

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Методические указания
по выполнению практических занятий и лабораторных работ
по дисциплине «ПРОГРАММНО-АППАРАТНЫЕ СРЕДСТВА ЗАЩИТЫ
ИНФОРМАЦИИ»
для студентов специальности 10.05.01 Компьютерная безопасность
специализация «Информационно-аналитическая и техническая экспертиза
компьютерных систем»

Ставрополь
2026

ВВЕДЕНИЕ

Целью изучения дисциплины "Программно-аппаратные средства защиты информации" является формирование набора общепрофессиональных, профессиональных и профессионально-специализированной компетенций студента по специальности 10.05.01 Компьютерная безопасность.

Учебно-методическое пособие содержит 16 практических занятий и 8 лабораторных работ, подлежащих выполнению студентами.

Каждое практическое занятие состоит из следующих разделов:

- наименование занятия;
- цель занятия;
- общие сведения/сценарий;
- необходимые ресурсы;
- теоретический материал;
- методические пояснения и рекомендации по выполнению практического занятия;
- содержание отчета.

Учебно-методическое пособие содержит список литературы, общий для всех занятий. Методические указания выступают в качестве информационного и практического источника и могут быть использованы для очного и дистанционного обучения.

Указания по технике безопасности при выполнении практических занятий

Требования безопасности перед началом работ

- ЗАПРЕЩАЕТСЯ: переодеваться, пользоваться огнем, курить, принимать пищу в лаборатории;
- убедиться в отсутствии видимых повреждений на рабочем месте;
- разместить на столе тетради, учебные пособия так, чтобы они не мешали работе на компьютере;
- включение компьютера производить только после получения допуска по выполняемой работе и разрешения преподавателя или сотрудника лаборатории.

Требования безопасности во время работы

- ЗАПРЕЩАЕТСЯ: присоединять или отсоединять кабели, трогать разъемы, провода и розетки, открывать системный блок, пытаться самостоятельно устранять неисправности в работе оборудования, приносить и запускать компьютерные игры;

- выполняя практическое занятие, студенты обязаны использовать только вычислительную технику, периферийное оборудование, соединительные кабели, измерительное оборудование и носители информации, непосредственно относящиеся к данной работе;

- подключение и отключение составляющих вычислительного комплекса производить только при полном снятии напряжения со всех составляющих вычислительного комплекса;

- при обнаружении неисправностей в оборудовании немедленно отключить источники питания и доложить об этом руководителю занятий или сотруднику лаборатории;

Требования безопасности по окончании работы

- доложить руководителю занятий или сотруднику лаборатории о завершении работ;

привести в порядок и сдать рабочее место сотруднику лаборатории, и доложить руководителю занятий о сдаче.

1. Лабораторная работа №1

Анализ задач управления АИС и информационной безопасностью

Скажите мне честно - есть ли хоть какой-то выход из этого кошмара?

- Выход всегда есть, - ответил Эркюль Пуаро.

А. Кристи «Подвиги Геккулеса»

Задание:

1. Подготовиться к публичному собеседованию по материалам лекции 1;
2. По представленным материалам подготовить анализ

- ❖ структуры автоматизированной информационной системы (АС);
- ❖ компонентов информационной инфраструктуры;
- ❖ обобщенной архитектуры системы управления АС;
- ❖ задач управления системой информационной безопасности предприятия, выделив абонентскую и сетевую компоненты АС

Вводная часть

Ввиду растущей сложности современного бизнеса компаниям приходится постоянно внедрять все новые и новые технологии, устанавливая более мощное и качественное оборудование. Для успешного использования современных информационных технологий необходимо надежное и эффективное управление не только самими сетями, но и средствами сетевой безопасности. И если в прошлом задача заключалась в управлении отдельными серверами, сетями и маршрутизаторами, то сейчас требуется обеспечить информационную безопасность корпоративных бизнес-процессов (систем). В настоящее время на первый план выходит задача создания комплексной системы управления, которая охватывает всю инфраструктуру компании и, независимо от сложности и масштаба АС, позволяет:

- централизованно и оперативно оказывать управляющие воздействия на всю информационную инфраструктуру;
- проводить регулярный аудит и всеохватывающий мониторинг, дающие объективную информацию о состоянии информационной безопасности для принятия оперативных решений;
- накапливать статистические данные о работе информационной инфраструктуры для прогнозирования ее развития.

Основная задача любой информационной системы - создание условий для работы и ведения бизнеса, облегчение и автоматизация труда людей.

2. **Письменный контроль** основных определений понятий дисциплины.

3. **Собеседование** по вопросам лекции 1: см. Л 1;

4. **Отработка задания 1:** см. Л 1;

5. **Отработка задания 2:**

5.1. Методология управления информационными системами ITIL

Согласно методологии ITIL (IT Infrastructure Library), предложенной британским Центральным компьютерным и телекоммуникационным агентством (ССТА), информационная система состоит из двух крупных блоков: информационной инфраструктуры и информационных сервисов (рис. 1).



Рис. 1. Информационная система с точки зрения методологии ITIL

Информационные системы создаются для обеспечения условий функционирования информационных сервисов. Это могут быть Internet-сервисы, сервисы приложений, сервисы управления, принятия решений и т.п. Однако ни один сервис невозможно себе представить без информационной инфраструктуры.

Информационная инфраструктура является материальной основой, той средой, в которой функционируют информационные сервисы. Она представляет собой совокупность технических средств, линий связи, процедур, нормативных документов и т.п., которые обеспечивают функционирование сервисов. Качество информационных сервисов напрямую зависит от качества информационной инфраструктуры и управления ею.

Информационную инфраструктуру можно представить, как пирамиду (рис. 2), в основании которой лежат информационные ресурсы (вычислительные платформы, серверы, ПК, сети передачи данных, линии связи). Второй уровень пирамиды образуют различные приложения. Они используют ресурсы из первого

слоя и обеспечивают работу сервисов таких конкретных приложений, как прикладное ПО, электронная почта, системы гарантированной доставки, базы данных, Web-сервисы и т.д. И наконец, на самом верхнем уровне функционируют приложения, обеспечивающие прохождение бизнес- и производственных процессов. Эти приложения, использующие оба нижних слоя, направлены на решение таких бизнес-задач, как управление производством, взаимодействие с заказчиками и поставщиками, финансовый учет и поддержка принятия решений.

Методология ITIL изложена в наборе всесторонних и согласованных документов, построенных на основе знаний и опыта мировых организаций, и предназначена для управления обслуживанием ИС. Библиотека ITIL состоит из более чем 40 книг, разработанных британским Центральным компьютерным и телекоммуникационным агентством (ССТА). Библиотека была создана в конце 1980-х годов, когда многие организации оказались все более зависимыми от ИС.



Рис. 2. Составляющие информационной инфраструктуры

Цель методологии ITIL - повысить эффективность ИС в выполнении бизнес-задач и уменьшить затраты на предоставление или модернизацию сервисов ИТ. Концепция ITIL базируется на лучшей практике и опыте ведущих экспертов, консультантов, инженеров. На сегодня это наиболее целостный и полный документ, описывающий организацию управления обслуживанием ИС, своего рода стандарт де-факто.

Одним из первых способов управления сетевыми ресурсами было применение удаленного доступа к компьютеру или к иному сетевому устройству (например, к принтеру или маршрутизатору) с целью его администрирования, однако за прошедшие двадцать лет появились более сложные инструменты управления сетями. В общем случае архитектура системы управления сетью имеет вид, представленный на рис. 3. Приложение для управления сетью может выполняться на рабочем месте администратора сети или на другом компьютере.

Его назначение - собирать сведения об управляемых устройствах, которые поступают от так называемых *агентов* - приложений или сервисов операционной системы, выполняющихся на управляемых устройствах.

Обычно приложение для управления сетью оперирует большими объемами подобных данных и отображает их в том или ином виде с помощью пользовательского интерфейса консоли управления, доступной администратору сети, обрабатывая происходящие в сети события. Нередко подобные приложения могут выполнять определенные операции на основе собранных ими данных, например запускать тесты, останавливать или переконфигурировать управляемые устройства, выдавать диагностические сообщения.

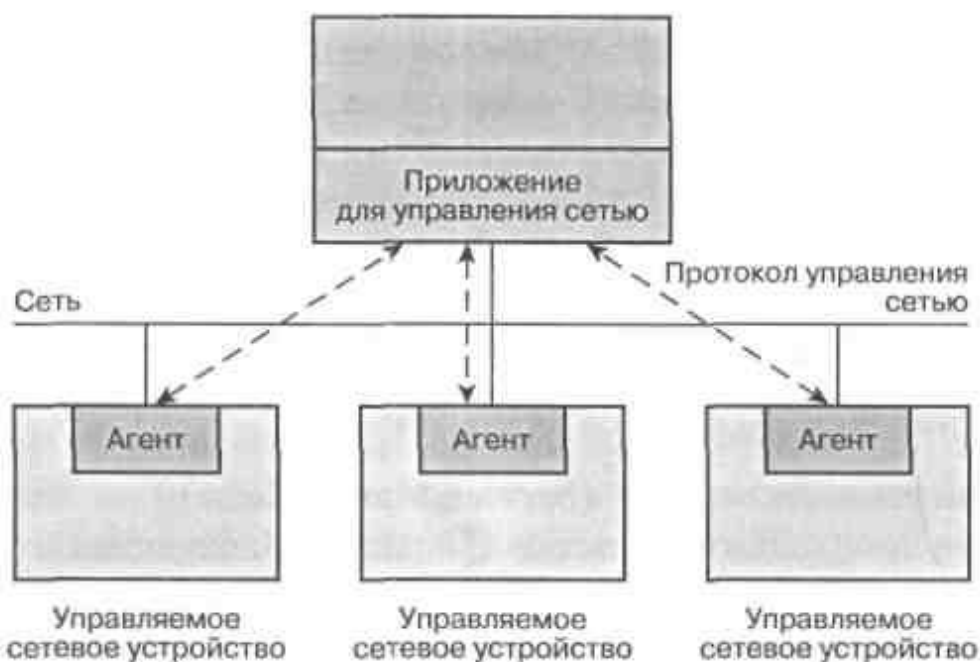


Рис. 3. Обобщенная архитектура системы управления сетью

Для взаимодействия подобных приложений с агентами обычно применяются открытые сетевые протоколы, такие как SMNP (Simple Network Management Protocol) - обычно в локальных сетях - или CMIP (Common Management Information Protocol) - обычно в случае распределенных сетей, использующих телекоммуникации, хотя некоторые производители программного обеспечения для управления сетями пользуются и собственными сетевыми протоколами.

Современные средства управления сетями могут выполнять следующие задачи:

- отслеживание сбоев в управляемых компьютерах и устройствах, определение и устранение их причин (нередко автоматическое), исправление их последствий и предотвращение сбоев (например, путем выполнения диагностических операций);
- управление конфигурированием компьютеров и сетевых устройств (в частности,

инициализация, переконфигурирование и выключение управляемых сетевых устройств и компьютеров);

- управление потреблением сетевых ресурсов пользователями и группами пользователей (например, регулирование дисковых или иных квот);
- управление производительностью сетевых устройств и сервисов (с помощью сбора и анализа статистики интенсивности применения и частоты ошибок сетевых устройств и искусственной установки уровня их производительности на основе полученных данных);
- управление защитой данных с помощью контроля доступа к сетевым ресурсам на основе заранее установленной политики безопасности и уведомления администратора о попытках ее нарушения.

Методология ITIL логически группирует наиболее популярные процессы в два набора, представляющие основные элементы управления ИТ-сервисами: сервисную поддержку (Service Support) и предоставление сервиса (Service Delivery).

Набор сервисной поддержки включает:

- управление конфигурацией (Configuration Management);
- взаимодействие с пользователями (Service Desk);
- управление проблемами (Problem Management);
- управление изменениями (Change Management);
- управление разработкой и распространением ПО (Software Control & Distribution).

Набор предоставляемого сервиса включает:

- управление уровнем сервиса (Service Management);
- управление производительностью (Capacity Management);
- управление доступностью (Availability Management);
- управление затратами (Cost management);
- управление непрерывностью (Contingency Management).

Многие организации используют ITIL в своей работе, поскольку эта концепция предлагает систематический и профессиональный подход к управлению информационными системами.

Задачи управления системой информационной безопасности предприятия. Важнейшим компонентом системы управления корпоративной сетью является система информационной безопасности предприятия. Сформулируем основные задачи управления системой информационной безопасности масштаба предприятия.

Система управления средствами защиты информации в распределенной сети масштаба предприятия должна решать следующие задачи:

- управление политикой безопасности в рамках сети предприятия, формирование локальных политик безопасности (ЛПБ) отдельных устройств и доведения ЛПБ до всех устройств защиты информации;
- управление конфигурацией объектов и субъектов доступа; включает управление составом, версиями, компонентами устройств и ПО защиты, а также управление патчами, которые служат для закрытия «дыр», обнаруженных в поставленных продуктах обеспечения безопасности;
- предоставление сервисов защиты распределенным прикладным системам, регистрация защищенных приложений и их ресурсов. Приложения этой группы должны обеспечивать прежде всего интерфейс (API) для управления сервисами защиты со стороны прикладных систем;
- управление криптосредствами, в частности ключевое управление (ключевая инфраструктура). Ключевая инфраструктура должна функционировать в составе инфраструктурных (системообразующих) служб;
- событийное протоколирование; включает настройку выдачи логов на разные устройства, управление уровнем детализации логов, управление составом событий, по которым ведется протоколирование;
- аудит безопасности информационной системы; обеспечивает получение и оценку объективных данных о текущем состоянии защищенности информационной системы; иногда под аудитом безопасности понимают анализ логов, поиск нарушителей и «дыр» в существующей системе; однако эти функции покрываются, скорее, задачами управления логами;
- мониторинг безопасности системы; обеспечивает получение информации в реальном времени о состоянии, активности устройств и о событиях с контекстом безопасности, происходящих в устройствах, например о потенциальных атаках;
- обеспечение работы специальных защищенных приложений, например нотариального надзора за операциями, а также поддержка регламентных мероприятий (смена ключей, паролей, устройств защиты, выпуск смарт-карт др.);
- обеспечение работы проектно-инвентаризационной группы приложений; эта группа приложений должна осуществлять:
 - ✓ определение точек установки средств защиты в сети предприятия;
 - ✓ учет применяемых средств защиты;
 - ✓ контроль модульного состава средств защиты;
 - ✓ контроль состояния средств защиты и др.

Существует проблема комплексирования и организации взаимодействия традиционных систем управления сетями и систем управления средствами защиты информации в сети. Для решения этой проблемы применяются два основных подхода.

Первый подход заключается в интеграции средств сетевого или системного управления с механизмами управления средств защиты. Средства сетевого и системного управления ориентированы в первую очередь на управление сетью или информационными системами, то есть поддерживают традиционные действия и услуги: управление учетными записями пользователей, управление ресурсами и событиями, маршрутизацию, производительность и т.п. Ряд компаний (в их числе Cisco Systems, Computer Associates, Hewlett Packard, PLATINUM Technology, Tivoli Systems) пошли по пути интеграции механизмов управления средств защиты в традиционные системы управления сетями. Однако часто такие комплексные системы управления отличаются высокой стоимостью, и кроме того, некоторые аспекты управления безопасностью остаются за пределами внимания этих систем.

Если использование таких комплексных решений в организации невозможно по каким-либо причинам, то имеет смысл обратить внимание на средства, предназначенные для решения только одной задачи - управления безопасностью. Например, Open Security Manager (OSM) от Check Point Software Technologies дает возможность централизованно управлять корпоративной политикой безопасности и устанавливать ее на сетевые устройства по всей компании. Продукт OSM является одной из основных компонент технологии OPSEC (Open Platform for Secure Enterprise Connectivity), разработанной компанией Check Point, он создает интерфейс для управления устройствами сетевой безопасности различных производителей (например, Cisco, Bay, 3Com).

2. Лабораторная работа №2

Реализация системы управления средствами безопасности

Структурно-продуктовая линия Trust Works подразделяется на агентов безопасности (Trusted Agent), центр управления (Trusted GSM Server) и консоль управления (Trusted GSM Console). Общая структурная схема решения показана на рис. 12.4.

Назначение основных средств безопасности

Агент безопасности, установленный на персональном компьютере клиента, ориентирован на защиту индивидуального пользователя, выступающего, как правило, клиентом в приложениях «клиент-сервер».

Агент безопасности, установленный на сервере приложений, ориентирован на обеспечение защиты серверных компонент распределенных приложений.

Агент безопасности, установленный на шлюзовом компьютере, обеспечивает развязку сегментов сети внутри предприятия или между предприятиями (business-to-business), решая задачи согласования политик безопасности разных сетей.

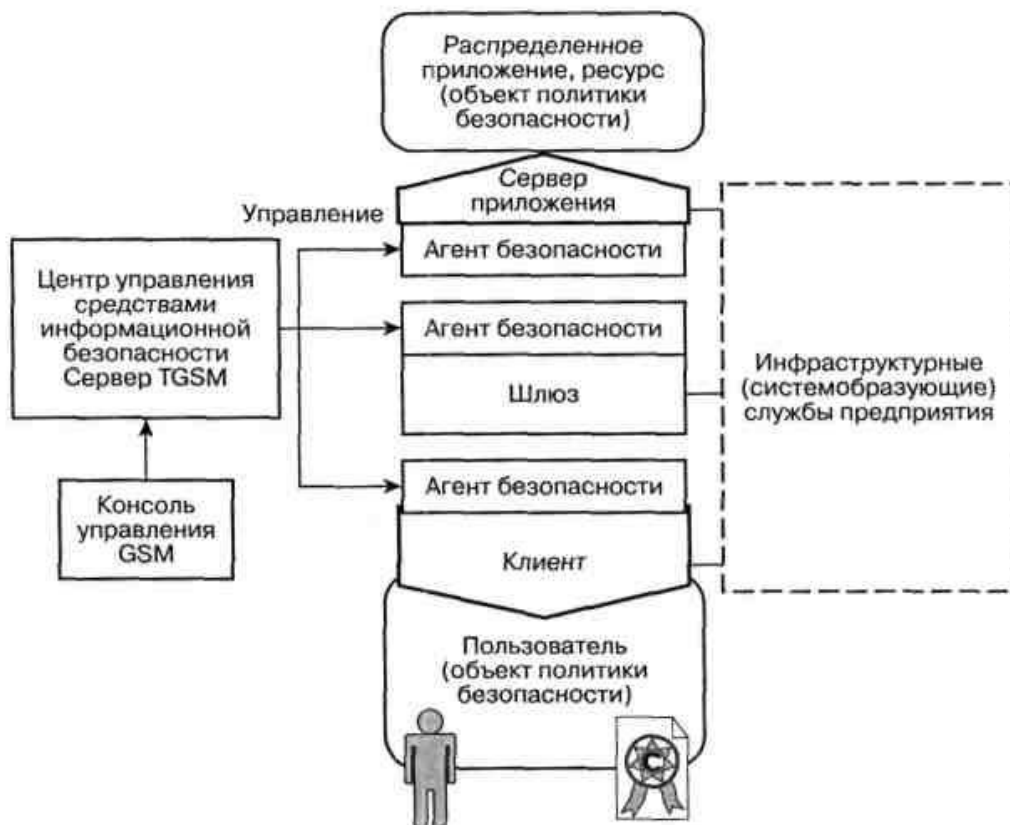


Рис. 12.4. Общая структурная схема системы управления средствами информационной безопасности

Центр управления обеспечивает описание глобальной политики безопасности в масштабах сети, трансляцию глобальной политики в локальные политики безопасности устройств защиты, загрузку устройств защиты и контроль состояний всех агентов системы. Для организации распределенной схемы управления без-

опасности предприятия в системе GSM предусматривается установка нескольких (до 65535) серверов GSM.

Консоль управления предназначена для организации рабочего места администратора (администраторов) системы. Для каждого из серверов GSM может быть установлено несколько консолей, каждая из которых настраивается согласно ролевым правам каждого администратора системы GSM.

Локальный агент безопасности представляет собой программу, размещаемую на оконечном устройстве (клиенте, сервере, шлюзе) и выполняющую следующие функции защиты:

- О аутентификацию объектов политики безопасности, включая интеграцию различных сервисов аутентификации;
- О определение пользователя в системе и событий, связанных с данным пользователем;
- О обеспечения централизованного управления средствами безопасности и контроля доступа;
- О управления ресурсами в интересах приложений, поддержку управления доступом к ресурсам прикладного уровня;
- О защиту и аутентификацию трафика;
- О фильтрацию трафика;
- О событийное протоколирование, мониторинг, тревожную сигнализацию.

Дополнительные функции локального агента безопасности (разрабатываются в составе решения GSM):

- О поставка криптосервиса (multiple concurrent pluggable modules);
- О управление периметрами Single Sign-On (как подзадача аутентификации пользователей);
- О сервис в интересах защищенных приложений (криптосервис, сервис доступа к PKI, доступ к управлению безопасностью);
- О сжатие трафика (IPcomp, pluggable module);
- О управление резервированием сетевых ресурсов (QoS);
- О функции локального агента сетевой антивирусной защиты.

Центральным элементом локального агента является процессор локальной политики безопасности (LSP processor), интерпретирующий локальную политику безопасности и распределяющий вызовы между остальными компонентами.

Защита ресурсов

Аутентификация и авторизация доступа

В рамках решения реализуется ряд различных по функциональности схем аутентификации, каждая из которых включает тип аутентификации и способ (механизм) идентификации объектов.

Для выбора типа аутентификации предусмотрены следующие возможности: аутентификация пользователя при доступе к среде GSM или локальной операци-

онной системе, аутентификация пользователя при доступе в сеть (сегмент сети), взаимная сетевая аутентификация объектов (приложение-приложение). Для выбора способа идентификации предусмотрены следующие варианты, обычно используемые совместно в любой конфигурации: токен (смарт-карта), пароль, «внешняя» аутентификация.

Контроль доступа при сетевых взаимодействиях

При инициализации защищенного сетевого соединения от локальной операционной системы или при получении запроса на установление внешнего соединения локальные агенты безопасности на концах соединения (и/или на промежуточном шлюзе) обращаются к ЛПБ устройства и проверяют, разрешено ли установление данного соединения. В случае, если такое соединение разрешено, обеспечивается требуемый сервис защиты данного соединения, если запрещено - сетевое соединение не предоставляется.

Контроль доступа на уровне прикладных объектов

Для незащищенных распределенных приложений в GSM обеспечивается сервис разграничения прав доступа на уровне внутренних объектов данного приложения. Контроль доступа на уровне объектов прикладного уровня обеспечивается за счет применения механизма проху Proxu разрабатывается для каждого прикладного протокола. Предусмотренным является протокол HTTP.

Для построения распределенной схемы управления и снижения загрузки сети в GSM используется архитектура распределенных проху-агентов (проху-модуль в составе локального агента), каждый из которых:

О имеет абстрактный универсальный интерфейс, обеспечивающий модульное подключение различных проху-фильтров; О имеет интерфейс к системе управления, но использует временный кэш для управления параметрами фильтрации; О осуществляет фильтрацию, управляемую обобщенными правилами типа:

- аутентифицировать субъект X в приложении-объекте Y;
- разрешить доступ субъекту X к объекту Y с параметрами P;
- запретить доступ субъекту X к объекту Z.

Семантика правил управления проху-фильтром и описания субъектов и объектов доступа зависят от конкретного прикладного протокола, однако центр управления имеет возможность регистрировать проху-фильтры и обеспечивать управление ими в контексте общей глобальной политики безопасности.

Проху-агент может быть установлен на шлюзе безопасности, непосредственно на сервере, исполняющем контролируемые приложения, и на клиентском месте системы.

Управление средствами защиты

Важнейшим элементом решения TrustWorks является централизованная, основанная на политике (policy based) система управления средствами сетевой и ин-

формационной безопасности масштаба предприятия. Эта система обеспечивает следующие качественные потребительские характеристики:

- О высокий уровень защищенности системы управления (путем выделения защищенного периметра управления внутри сети предприятия);
- О расширяемость системы управления информационной безопасностью;
- О высокий уровень надежности системы управления и ключевых ее компонентов;
- О интеграцию с корпоративными системами общего сетевого и информационного управления;
- О простую, интуитивно воспринимаемую, эргономичную и инфраструктурную среду описания, формирования, мониторинга и диагностики политики безопасности масштаба предприятия (enterprise level policy based management).

Управление осуществляется специальным программным обеспечением администратора - консолью управления. Количество и функции каждого из экземпляров установленной в системе ПО консоли управления задаются главным администратором системы в зависимости от организационной структуры предприятия. Для назначения функций каждого из рабочих мест консоли управления используется ролевой механизм разграничения прав по доступу к функциям управления (менеджмента) системы.

Функции управления GSM

В зависимости от вида управляемых объектов набор управляющих функций в GSM можно условно разбить на 3 категории:

- О управление информационным каталогом;
- О управление пользователями и правами доступа;
- О управление правилами ГПБ.

Функции управления информационным каталогом определяют информационную составляющую GSM:

- О формирование разделов каталога;
- О описание услуг каталога;
- О назначение и контроль сетевых ресурсов, требуемых для выполнения услуги;
- О регистрация описания услуги;
- О контроль состояния услуг или разделов каталога услуг;
- О мониторинг исполнения услуг;
- О подготовка и пересылка отчетов (протоколов) по состоянию каталога.

Для управления правами доступа пользователей системы к услугам (информационным или сетевым ресурсам) GSM выполняет следующие функции:

- О формирование групп пользователей по ролям и/или привилегиям доступа к услугам системы;
- О формирование иерархических агрегаций пользователей по административным, территориальным или иным критериям (домены и/или департаменты);
- О формирование ролей доступа пользователей к услугам (информационным или

сетевым ресурсам);

- О назначение уровней секретности для услуг и пользователей системы (поддержка мандатного механизма разграничения прав);
- О назначение фиксированных прав доступа группам, ролям, агрегациям пользователей или отдельным пользователям системы к информационным или сетевым ресурсам системы;
- О подготовка и пересылка отчетов (протоколов) по доступу пользователей к услугам системы;
- О подготовка и пересылка отчетов (протоколов) по работе администраторов системы.

Правила ГПБ ставят в соответствие конкретные свойства защиты (как для сетевых соединений, так и для доступа пользователей к информационным услугам) предустановленным уровням безопасности системы. Контроль за соблюдением правил ГПБ выполняет специальный модуль в составе сервера системы - процессор политики безопасности (Security Policy Processor), обеспечивающий следующие функции системы:

- О определение каждого из уровней безопасности набором параметров защиты соединений, схемы аутентификации и разграничения прав;
- О назначение уровней безопасности конкретным услугам или разделам каталога услуг;
- О назначение уровней безопасности пользователям или любым агрегациям пользователей системы (группам, ролям, доменам, департаментами)
- О контроль за целостностью ГПБ (полнотой правил);
- О вычисление ЛПБ локальных устройств защиты - агентов безопасности и контроль за их исполнением;
- О контроль за исполнением ГПБ по различным критериям;
- О подготовка и пересылка отчетов (протоколов) о состоянии системы и всех попытках нарушения ГПБ.

Каждый из администраторов системы аутентифицируется и работает с системой через консоль управления согласно предустановленным для него правам (на каталог ресурсов или его часть, на определенный ролями набор функций управления, на группы или другие наборы пользователей). Все действия любого администратора протоколируются и могут быть попарно контролируемы.

В целях обеспечения непротиворечивости с существующими у пользователя системами управления предприятия (Enterprise & Network Management System) и простоты управления в GSM реализуется тесная интеграция с наиболее популярными платформами управления (CA Unicenter TNG или HP 0V), выбор определяется менеджером продуктов.

Лабораторная работа №3

Технология анализа защищенности

Цель: Усвоить последовательность анализа защищенности автоматизированной системы при построении политики информационной безопасности

Задание: Проработать материал занятия и предыдущей лекции (тема 2 Л-1) и составить блок – схему технологии анализа защищенности в рамках построения политики информационной защищенности.

В организации, использующей корпоративную информационную систему, приходится регулярно проверять, насколько реализованные или используемые механизмы защиты информации соответствуют положениям принятой в организации политики безопасности. Такая задача периодически возникает при изменении и обновлении компонентов информационной системы, изменении конфигурации операционной системы и т.п.

Однако администраторы сетей не имеют достаточно времени на проведение подобных проверок для всех узлов корпоративной сети. Поэтому специалисты отделов защиты информации нуждаются в средствах, облегчающих анализ защищенности используемых механизмов обеспечения информационной безопасности. Этот процесс помогают автоматизировать средства анализа защищенности, называемые на Западе *сканерами безопасности* (security scanners).

Использование средств анализа защищенности позволяет определить уязвимости на узлах корпоративной сети и устранить их до того, как ими воспользуются злоумышленники. По существу, работа системы анализа защищенности аналогична действиям охранника, периодически обходящего все этажи охраняемого здания в поисках незапертых дверей, открытых окон и других «брешей». Только в качестве здания выступает корпоративная сеть, а в качестве незакрытых окон и дверей - уязвимости.

Если снова обратиться к рис. 1, поясняющему этапы развития атаки на корпоративную информационную систему, то можно заметить, что средства анализа защищенности работают на первом этапе осуществления атаки. Обнаруживая и своевременно устраняя уязвимости, они таким образом предотвращают саму возможность реализации атаки, что позволяет снизить затраты (финансовые, ресурсные, человеческие и т.д.) на эксплуатацию средств защиты.

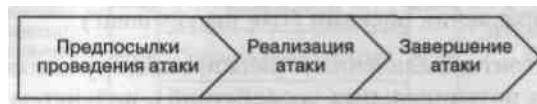


Рис. 1. Этапы осуществления атаки

Технологии анализа защищенности являются действенным методом, позволяющим проанализировать и реализовать политику сетевой безопасности прежде, чем осуществится попытка ее нарушения снаружи или изнутри организации.

Средства анализа защищенности могут функционировать на сетевом уровне, уровне операционной системы и уровне приложения. Они могут проводить поиск уязвимостей, постепенно наращивая число проверок и «углубляясь» в информационную систему, исследуя все ее уровни.

Наибольшее распространение получили средства анализа защищенности сетевых сервисов и протоколов. Обусловлено это, в первую очередь, универсальностью используемых протоколов. Изученность и повсеместное использование таких протоколов, как IP, TCP, HTTP, FTP, SMTP и т.п., позволяют с высокой степенью эффективности проверять защищенность информационной системы, работающей в сетевом окружении.

Вторыми по распространенности являются средства анализа защищенности операционных систем. Обусловлено это также универсальностью и распространенностью некоторых операционных систем (например, UNIX и Windows NT). Однако из-за того, что каждый производитель вносит в операционную систему свои изменения (в качестве примера можно привести множество разновидностей ОС UNIX), средства анализа защищенности ОС анализируют в первую очередь параметры, характерные для всего семейства данной ОС. Лишь для некоторых систем предусмотрен анализ их специфичных параметров.

Средств анализа защищенности приложений на сегодняшний день не так уж много. Такие средства пока существуют только для широко распространенных прикладных систем типа Web-браузеров (Netscape Navigator, Microsoft Internet Explorer), СУБД (Microsoft SQL Server, Oracle) и т.п.

Применение средств анализа защищенности позволяет быстро определить все узлы корпоративной сети, доступные в момент проведения тестирования, выявить все используемые в сети сервисы и протоколы, их настройки и возможности несанкционированного воздействия (как изнутри корпоративной сети, так и снаружи). По результатам сканирования эти средства вырабатывают рекомендации и пошаговые меры, позволяющие устранить выявленные недостатки.

Данный метод контроля нарушений политики безопасности не может

заменить специалиста по информационной безопасности. Средства анализа защищенности призваны лишь автоматизировать поиск некоторых известных уязвимостей. Взаимодействие абонентов в любой сети базируется на использовании сетевых протоколов и сервисов, определяющих процедуру обмена информацией между двумя и более узлами. При разработке сетевых протоколов и сервисов к ним предъявлялись требования (обычно явно недостаточные) по обеспечению безопасности обрабатываемой информации. Поэтому постоянно появляются сообщения об обнаруженных в сетевых протоколах уязвимостях. В результате возникает потребность в постоянной проверке всех используемых в корпоративной сети протоколов и сервисов.

Системы анализа защищенности выполняют серию тестов по обнаружению уязвимостей. Эти тесты аналогичны применяемым злоумышленниками при осуществлении атак на корпоративные сети.

Сканирование с целью обнаружения уязвимостей начинается с получения предварительной информации о проверяемой системе, в частности о разрешенных протоколах и открытых портах, используемой версии операционной системы и т.п. Заканчивается сканирование попытками имитации проникновения; используются широко известные атаки, например подбор пароля методом полного перебора (brute force - «грубая сила»).

При помощи средств анализа защищенности на уровне сети можно тестировать не только возможность несанкционированного доступа в корпоративную сеть из Internet. Эти же средства могут быть использованы для аналогичной проверки во внутренней сети организации. Системы анализа защищенности на уровне сети могут быть использованы как для оценки уровня безопасности организации, так и для контроля эффективности настройки сетевого программного и аппаратного обеспечения.

В настоящее время известно более десятка различных средств, автоматизирующих поиск уязвимостей сетевых протоколов и сервисов. Одно из первых таких средств - свободно распространяемая система анализа защищенности UNIX-систем SATAN (Security Administrator Tool for Analyzing Networks), разработанная В. Венема и Д. Фармером. Среди коммерческих систем анализа защищенности можно назвать Internet Scanner компании Internet Security Systems Inc., NetSonar компании Cisco, CyberCop Scanner компании Network Associates и ряд других.

Типичная схема проведения анализа защищенности (на примере системы Internet Scanner) представлена на рис. .

Средства анализа защищенности данного класса анализируют уязвимость не только сетевых сервисов и протоколов, но и системного и прикладного программного обеспечения, отвечающего за работу с сетью. К такому обеспечению можно отнести Web-, FTP- и почтовые серверы, межсетевые

экраны, браузеры и т.п.

Некоторые из предлагаемых на рынке систем проводят не только анализ защищенности программного обеспечения, но и сканирование аппаратных средств. К таким аппаратным средствам относится коммутирующее и маршрутизирующее оборудование.

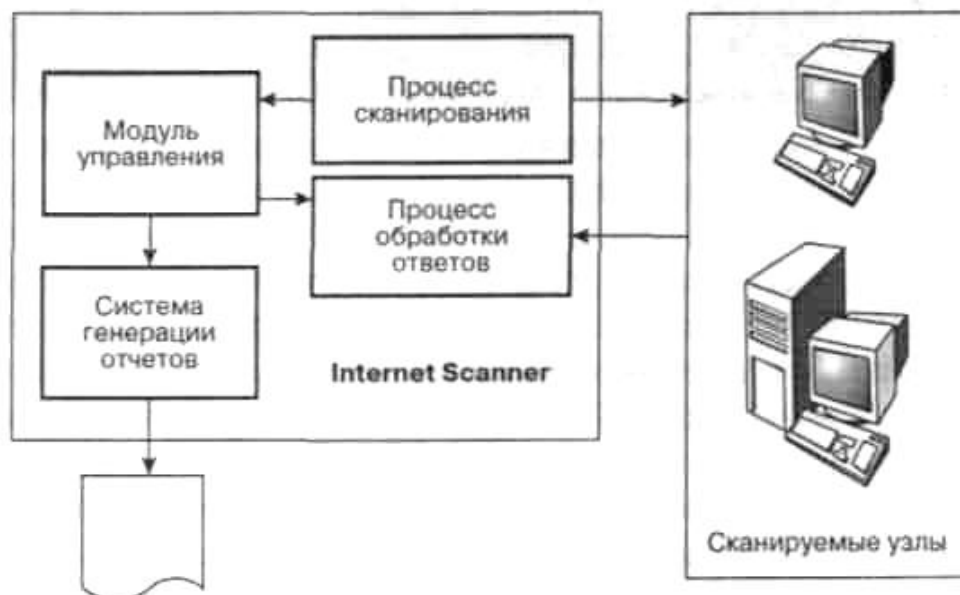


Рис. 2. Схема проведения анализа защищенности
(на примере системы Internet Scanner)

Средства этого класса предназначены для проверки настроек операционной системы, влияющих на ее защищенность. К таким настройкам можно отнести:

О учетные записи пользователей (accounts), например длину пароля и срок его

действия; О права пользователей на доступ к критичным системным файлам; О уязвимые системные файлы; О установленные патчи (patches) и т.п.

Системы анализа защищенности на уровне ОС также могут быть использованы для контроля конфигурации операционных систем.

В отличие от средств анализа защищенности сетевого уровня, эти системы проводят сканирование не снаружи, а изнутри анализируемой ОС, то есть не имитируют атаки внешних злоумышленников. Кроме обнаружения уязвимостей некоторые системы анализа защищенности на уровне ОС (например, System Scanner компании Internet Security Systems) автоматически устраняют ряд обнаруженных проблем или корректируют параметры системы, не удовлетворяющие политике безопасности, принятой в организации.

Как уже отмечалось, средств анализа защищенности операционной системы существует меньше, чем средств проверки сети. Среди первых

известных систем можно назвать COPS (Computerized Oracle and Password System), разработанную одним из авторов системы SATAN Д. Фармером совместно с Ю. Спаффордом из лаборатории COAST. Также широко известны системы System Security Scanner и Security Policy Manager компании Internet Security Systems Inc., Kane Security Analyst компании Intrusion Detection и др. Остановимся на некоторых общих характеристиках и требованиях к средствам анализа защищенности, которые нужно учитывать при их выборе и приобретении.

Обязательное условие, которое предъявляется к выбираемой системе, - отсутствие необходимости изменения сетевой инфраструктуры предприятия. В противном случае затраты на такую реорганизацию могут превысить стоимость самой системы анализа защищенности. В настоящий момент только разработка Internet Scanner компании Internet Security Systems удовлетворяет данному требованию.

При неправильном применении средств анализа защищенности ими могут воспользоваться злоумышленники для проникновения в корпоративную сеть. Поэтому средства анализа защищенности должны быть обеспечены механизмами разграничения доступа к своим компонентам и собранным данным.

К числу таких механизмов можно отнести:

- ограничение на запуск данных средств только пользователем с правами администратора; О шифрование архивов данных сканирования; О аутентификацию соединения при удаленном управлении; О установку специальных прав на каталоги с системой и т.п. Необходимо обратить внимание на наличие следующих возможностей процесса обнаружения уязвимостей:
- увеличения скорости сканирования за счет параллельной обработки нескольких устройств или сервисов;
- посылки уведомлений на каждый сканируемый узел сети во избежание несанкционированного использования системы;
- настройки на эксплуатационные требования сети для минимизации ложных срабатываний.

Постоянные изменения состояния корпоративной сети влияют и на ее защищенность. Поэтому в хорошей системе анализа защищенности должен быть предусмотрен режим работы по расписанию, чтобы, не дожидаясь, пока администратор «вспомнит» о ней, она самостоятельно проверяла уязвимости узлов сети и не только оповещала администратора о возникших проблемах, но и «рекомендовала» способы устранения выявленных уязвимостей. Если же такая возможность отсутствует в самой системе анализа защищенности, то необходимо поинтересоваться у поставщика, может ли предлагаемая система работать из командной строки. Подобный режим позволит воспользоваться

встроенными в операционную систему сервисами, позволяющими запускать заданные программы по расписанию (в качестве примера приведем службу расписания АТ для ОС Windows NT или CRON для ОС UNIX). Одной из важных характеристик, на которую необходимо обратить внимание, является наличие системы генерации отчетов. Независимо от эффективности используемых механизмов обнаружения уязвимостей средства анализа защищенности вряд ли найдут применение в организации, если в них отсутствует система генерации отчетов. Она должна позволять создавать документы различной степени детализации и для разных категорий пользователей, начиная с технических специалистов и заканчивая руководителями организации.

Форма представления данных в отчетах имеет немаловажное значение. Документ, насыщенный только текстовой информацией, не принесет пользы. Использование же графики наглядно продемонстрирует руководству все проблемы с безопасностью в сети организации. Обязательное условие при выборе средств анализа защищенности - наличие в отчетах рекомендаций по устранению найденных проблем.

Постоянное обнаружение новых уязвимостей требует, чтобы система анализа защищенности предусматривала возможность пополнения базы данных уязвимостей.

Это может быть реализовано как при помощи специального языка описания уязвимостей (например, APX в системах компании ISS, CASL в системе CyBeгCor Scanner), так и путем периодического пополнения уязвимостей, обеспечиваемого производителем системы. Для последующего анализа изменений уровня защищенности узлов корпоративной сети выбираемое средство должно позволять накапливать сведения о проведенных сеансах сканирования.

Эффективность применения систем анализа защищенности будет достигнута только в том случае, если их производитель постоянно обеспечивает свои продукты соответствующей поддержкой, учитывающей последние изменения в области обеспечения информационной безопасности. Информация об уязвимостях должна постоянно пополняться. Например, система SATAN не может быть рекомендована как надежное и эффективное средство, так как последние изменения в нее вносились несколько лет назад.

Однако следует отметить, что информация об уязвимости обычно появляется в базе данных системы не раньше, чем найдется ей «противоядие». Это является определенным недостатком всех способов поиска уязвимостей, поскольку злоумышленники могут воспользоваться имеющейся информацией до того, как появится способ устранения найденной уязвимости.

3. Лабораторная работа № 4

Постановка задач синтеза систем в конфликтных условиях как минимаксная задача

При постановке задачи синтеза системы (процесса, алгоритма, стратегии,...) полагаются заданными [1-12]: множество X рассматриваемых в задаче объектов (допустимых вариантов построения систем), множество Y условий функционирования систем и функционал $Q: X \times Y \rightarrow R$ (где R - множество вещественных чисел), определяющий значение $Q\{X, Y\}$ обобщенного показателя эффективности (ОПЭ) для любых элементов $X \in X$ и $Y \in Y$. Процедура задания конструкций X , Y и $Q\{\cdot\}$ и их интерпретация в терминах вариантов построения и характеристик реальных объектов составляют неформальную сторону модели класса исследуемых систем. Формальная часть модели заключается в дальнейшем использовании данных конструкций для постановки и решения математических задач синтеза систем.

Классическая задача синтеза формулируется при задании элемента $Y_0 \in Y$, определяющего (конкретизирующего) условия (среду) функционирования синтезируемой системы. Она заключается в максимизации ОПЭ $Q\{X, Y_0\}$ для заданных условий $Y = Y_0$ функционирования системы элементами $X \in X$ и записывается в виде

$$Q\{X, Y_0\} \rightarrow \max_{X \in X} \quad (1.1)$$

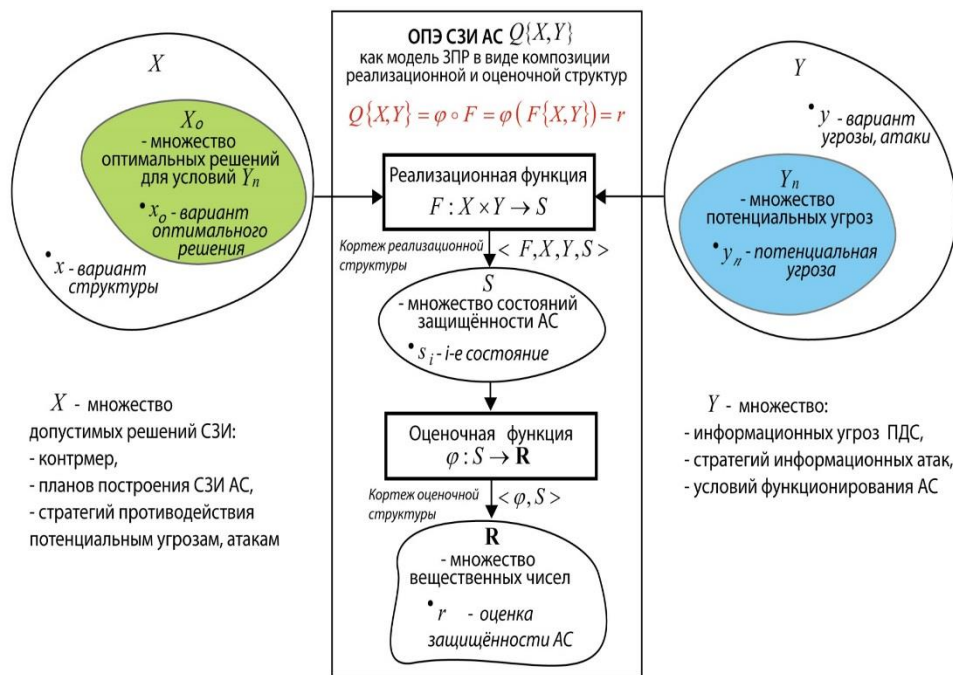
В качестве решений задачи (1.1) рассматривается множество $X_e(X, Y_0)$ так называемых e -оптимальных систем $X_e (e \geq 0)$, определяемое выражением.

$$X_e \in X_e(X, Y_0) \Leftrightarrow Q\{X, Y\} \leq Q\{X_e, Y_0\} + e$$

для любых $X \in X$.

Другими словами, изменение варианта построения системы относительно e -оптимального не может повысить эффективность ее функционирования в условиях Y_0 более чем на величину e . При этом необходимость рассмотрения e -оптимальных решений для случая $e > 0$ обусловлена как целесообразностью упрощения методов их поиска и нахождения просто реализуемых решений, так и принципиальным несуществованием оптимальных элементов (при $e = 0$) для определенных конструкций X и $Q\{X, Y_0\}$.

Концептуальная структура ЗПР при синтезе защищенной АС в условиях информационного конфликта



В последнем случае $X_0(X, Y_0) = \emptyset$ и задача поиска оптимального элемента по существу теряет смысл.

Из определения системы в условиях неопределенности полагается неполное задание условий функционирования проектируемой системы. Фактически в этом случае определяется целый класс сред (например, множество Y), в которых может применяться система. При этом для формулировки математической корректной задачи, так или иначе, используются дополнительные сведения об условиях применения системы, содержание которых и порождает различные подходы к ее синтезу.

Важным свойством класса условий Y , приводящим к специфическим задачам синтеза систем, является их конфликтность. Сущность конфликтной среды заключается в том, что конкретизация элемента $Y \in Y$ для данной системы X может осуществляться противной стороной (противником), целью которой является решение противоположной задачи (т.е. уменьшение значения ОПЭ $Q\{\}$). Целесообразность рассмотрения задачи синтеза в данном случае обусловлена практической необходимостью проектирования систем, функционирование которых предполагается при применении противником средств, специально разработанных для подавления проектируемой системы.

Пусть, например, для функционирования в конфликтных условиях, определенных классом Y , разработана и внедрена система $X \in X$. При этом решение задачи оптимизации противодействия

$$Q\{X, Y\} \rightarrow \min_{Y \in Y} \quad (1.2)$$

обеспечит противной стороне возможность подавления системы X до уровня, определяемого значением опэ, сколь угодно близким к величине

$$Q\{X, Y\} \triangleq \inf_{Y \in Y} Q\{X, Y\}.$$

При построении корректной модели класс Y должен характеризовать всю совокупность возможностей, реально имеющихся у противника по выбору способов противодействия синтезируемой системе, а функционал $Q\{\}$ - учитывать, кроме того, предпочтительность такого выбора. Тогда, очевидно, величина $Q\{X, Y\}$ определяет эффективность применения системы $X \in X$ и задача синтеза системы в конфликтных условиях формулируется, таким образом, как минимаксная задача вида:

$$Q\{X, Y\} = \inf_{Y \in Y} Q\{X, Y\} \rightarrow \max_{X \in X} \quad (1.3)$$

В КАЧЕСТВЕ РЕШЕНИЙ ЗАДАЧИ (1.3) РАССМАТРИВАЕТСЯ МНОЖЕСТВО $X_e(X, Y)$ e -оптимальных систем X_e ($e \geq 0$), определяемое выражением

$$X_e \in X_e(X, Y) \Leftrightarrow \inf_{Y \in Y} Q\{X, Y\} \leq \inf_{Y \in Y} Q\{X_e, Y\} + e \quad (1.4)$$

для любых $X \in X$.

Другими словами, изменение варианта построения систем относительно e -оптимального не может повысить эффективность ее функционирования в условиях оптимизированного противодействия более, чем на величину e .

Из определения e -оптимальной системы следует, что при ее применении в классе условий Y гарантируется значение ОПЭ, равное:

$$d_0\{X_e\} = \inf d\{X_e, Y\} \geq Q\{X, Y\} - e \quad (1.5)$$

где

$$Q\{X, Y\} \triangleq \sup_{X \in X} \inf_{Y \in Y} Q\{X, Y\}$$

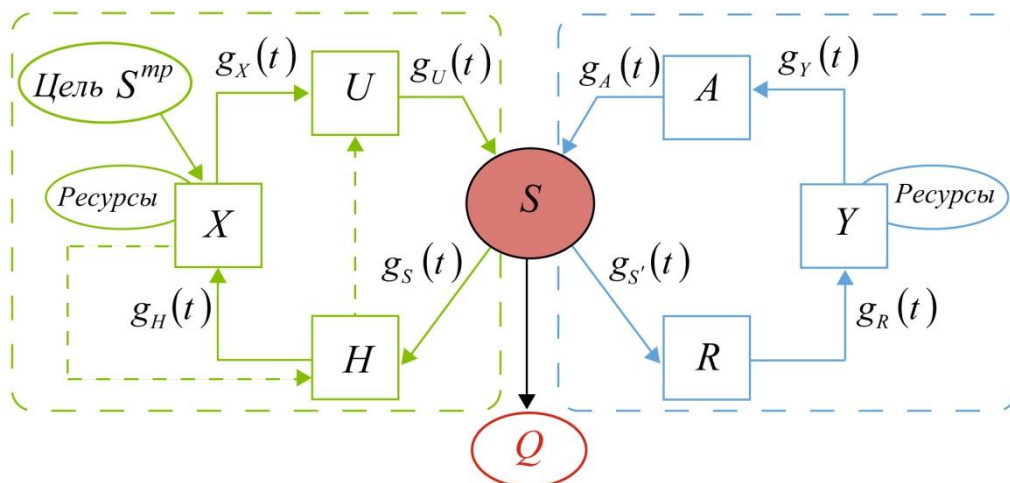
4. Лабораторная работа № 5

Элементы обобщенной модели взаимодействия ПДС

Для содержательной постановки и решения задач синтеза систем связи в условиях оптимизированного противодействия необходимы конкретизация структуры конструкций X , Y , $Q\{\cdot\}$ и их интерпретация в терминах вариантов построения и характеристик реальных объектов. С этой целью процесс функционирования системы при воздействии на нее подсистемы управления, противодействующей системы, а также совокупности случайных факторов и описывается обобщенной динамической моделью взаимодействия систем, представленной функциональной схемой на рис.1.

Обобщенная модель взаимодействия конфликтующих систем (макрообъекта W)

Компоненты обобщенной модели макрообъекта W



$$W = \langle U, A, S, H, R, X, Y \rangle$$

S - управляемый объект - ЗАС,

H - канал наблюдения,

R - канал разведки,

U - канал управления,

Y - СПР на подавление,

X - СПР на управление,

A - канал подавления,

Q – обобщенный показатель эффективности ЗАС.

В соответствии с данной моделью макрообъект, описывающий совокупность взаимодействующих элементов СВС и СП, представляется в виде композиции элементов: S - управляемого объекта (УО), X - системы (подсистемы) принятия решений на управление (СПРУ), H - канала наблюдения (КН), U - канала управления (КУ), Y - системы принятия решений на подавление (СПРП), R - канала разведки (КР) и V - канала подавления (КП).

Задача построения управляемой системы рассматривается как задача синтеза стратегии (оператора) управления ее состоянием. При этом к СПРУ и СПРП относятся все неизвестные (искомые) в задаче синтеза системы элементы, а остальные объекты

модели (УО, КН, КУ, КР, КП) задают имеющиеся ограничения на реализуемые процессы управления и подавления, а также воздействия этих процессов на состояние УО.

УО представляет собой совокупность всех тех элементов системы:

- на управление (синтез стратегии управления) которыми направляется задача;
- наблюдение за которыми целесообразно учитывать при выработке решения на управление состояниями системы;
- наблюдение за которыми может оказаться полезным СПРП для выработки решения на подавление системы.

СПРУ представляет собой совокупность всех органов, выполняющих функции принятия решений на управление УО или его отдельными элементами.

КН является подсистемой, обеспечивающей сбор сведений СПРУ о состоянии УО,

КУ представляет совокупность объектов, осуществляющих доведение распоряжений СПРУ, и исполнительных органов.

СПРП, КР и КП являются элементами противодействующей системы, выполняющими функции, аналогичные функциям СПРУ, КН и КУ.

Кроме того, отдельные из введенных объектов могут представляться в виде композиции составляющих элементов, что позволяет более просто задавать и интерпретировать их описания. В этом отношении существенной является декомпозиция СПРУ по подсистемам принятия решений различных уровней иерархии, которая в задачах синтеза обеспечивает задание дополнительных ограничений на класс допустимых стратегий принятия решений для предварительно определенных структур СПРУ.

В ряде случаев могут оказаться целесообразными декомпозиция КУ с выделением из него элементов, определяющих зависимость реализуемого управления от состояния УО (пунктирная линия на рис.1), а также выделение из КП элементов, характеризующих ресурс противодействующей системы и определяющих, таким образом, ограничения на реализуемые воздействия СПРП. Вместе с тем следует отметить, что с позиций решения задачи соответствующие модификации являются непринципиальными, поскольку все заданные особенности построения СПРУ, КН, КУ, СПРП, КР и КП могут быть учтены в ограничениях на классы допустимых решений СПРУ и СПРП [2,3, 5,6] . По сути дела все задачи, формулируемые на представленной модели, допускают адекватную постановку при исключении из рассмотрения КН, КР, КУ и КП: выделение данных элементов, является удобным лишь с точки зрения интерпретации конструкций X и Y .

Пусть задано множество T моментов времени, в которые наблюдается макрообъект ($\inf T = t_0$, где t_0 - начальный момент времени), и значения процессов $g_E(t)$, $E \in V \triangleq \{U, V, S, H, R, X, Y\}$ на выходах соответственно КУ, КП, УО, КН, КР, СПРУ, СПРП являются элементами определенных множеств G_E

Каждый объект $E \in V$ в общем случае описывается соответствующим причинным оператором [10,14,15] $M_E = \langle M_E^t \rangle, t \in T$, вырабатывающим по совокупности входных процессов $\{g_{E'}(r)\}$, $r \in T_t$, $E' \in V$ (где $T_t = T \cap [t_0, t]$)

некоторую характеристику $m_E(t)$, определяющую способ конкретизации значения $g_E(t)$ в виде

$$m_E(t) = M_E^t[\{g_{E'}(r)\}, r \in T_t, E' \in V]. \quad (2.1)$$

В соответствии с представленной схемой можно записать уравнения, задающие поведение макрообъекта $W = \langle U, V, S, H, R, X, Y \rangle$ для любых $t \in T$

$$\begin{aligned} m_U(t) &= M_U^T[g_U(r), g_X(r), r \in T_t], \quad m_V(t) = M_V^T[g_V(r), g_Y(r), r \in T_t], \\ m_S(t) &= M_S^T[g_U(r), g_V(r), r \in T_t], \quad m_H(t) = M_H^T[g_H(r), g_S(r), r \in T_t], \\ m_R(t) &= M_R^T[g_R(r), g_S(r), r \in T_t], \quad m_X(t) = M_X^T[g_X(r), g_H(r), r \in T_t], \\ m_Y(t) &= M_Y^T[g_Y(r), g_R(r), r \in T_t]. \end{aligned} \quad (2.2)$$

При этом процессы $g_E(t)$ $E \in V$ интерпретируются следующим образом: $g_U(t)$, $g_V(t)$ - реализованные воздействия на УО (систему связи) подсистемы управления и системы подавления ($g_U(t)$ - управление, $g_V(t)$ - подавление); $g_S(t)$ - состояние УО; $g_H(t)$, $g_R(t)$ - наблюдения состояния УО (СПРУ и СПРП); $g_X(t)$, $g_Y(t)$ - решения СПРУ и СПРП (на управление и подавление).

Оператор M_S полагается безынерционным, поскольку инерционности процессов управления и подавления синтезируемой (анализируемой) системы учитываются в КУ и КП. Более того, не нарушая общности рассмотрения, процесс на выходе УО может представляться в виде сочленения процессов $g_U(t)$ и $g_V(t)$: $g_S(t) = (g_U(t), g_V(t))$. Введение здесь для УО специального оператора M_S также обусловливается соображениями удобства интерпретации.

СОДЕРЖАНИЕ

Практическая работа № 1.....	3
Практическая работа № 2.....	14
Практическая работа № 3.....	25
Практическая работа № 4.....	29
Практическая работа № 5.....	35
Практическая работа № 6.....	47
Практическая работа № 7.....	54
Практическая работа № 8.....	58
Приложения.....	75

Практическая работа №1

Изучение блоков **generate, start, terminate, transfer, loop, displace, advance.**

Изучение генератора псевдослучайных чисел

Цель работы: знакомство и работа в системе GPSS с основными блоками

Формируемые компетенции: ПК-1, ПК-2, ПК-3, ПК-7, ПК-37

Теоретическая часть

1. GENERATE

GENERATE – блок, через который транзакты входят в модель. Имеет следующий формат записи:

GENERATE [A],[B],[C],[D],[E]

Скобки [] означают, что операнд является необязательным. Не существует ограничений на число различных блоков в одной модели.

Интервалы времени между последовательными появлениями транзактов блока GENERATE называют интервалом поступления. Все разработчики должны задавать спецификацию распределения интервалов времени поступления в блоке GENERATE. Информация, необходимая для этого, задаётся операндами А и В. Все возможные виды распределения интервалов времени поступления делят на *равномерно распределённые* и все *другие виды распределения*.

Операнд А – средний интервал времени между последовательными поступлениями транзактов в модель.

Операнд В – задаёт модификатор, который изменяет значения интервала генерации транзактов по сравнению с интервалом, указанным операндом А. Есть два типа модификаторов: модификатор – интервал и модификатор – функция.

Операнды А и В могут быть именем, положительным числом, выражением в скобках или непосредственно СЧА.

С помощью *модификатора – интервала* задаётся *равномерный закон* распределения времени между генерацией транзактов. При вычислении разности значений (А-В), заданных операндами А и В, получается нижняя граница интервала, а при вычислении суммы (А+В) – верхняя граница. После генерации очередного транзакта выбирается число из полученного интервала, это и будет значение времени, через которое следующий транзакт выйдет из блока GENERATE. Предположим, что транзакт входит в модель – блок GENERATE – в момент модельного времени 25.6. После того, как этот транзакт попадает в следующий блок модели, планировщик GPSS разыгрывает случайное значение из распределения интервалов времени, равного 8.4 ± 4.3 . Пусть разыгранным значением будет число 9.7. Тогда планировщик планирует приход следующего транзакта в блок GENERATE в момент времени $25.6 + 9.7 = 35.3$. Можно выбрать для розыгрыша генератор равномерно распределённых случайных чисел. Это устанавливается на странице Random Numbers в журнале настроек модели (Edit→Settings→Random Numbers), на которой в поле ввода GENERATE ввести номер генератора – любое положительное число. По умолчанию используется генератор номер 1.

Когда операнды А и В задают в виде констант (В – модификатор – интервал), они должны быть выражены неотрицательными числами. Также они не обязательно должны быть заданы, в таком случае предполагается их нулевое значение. В таком случае будет задано детерминированное значение интервалов времени.

Более сложные интервалы времени поступления транзактов (не по равномерному закону) могут быть заданы с использованием *модификатора – функции*. Под действием модификатора – функции значение операнда А умножается на значение функции, заданной операндом В. *При любом способе вычисления интервала времени значение операнда В не должно превышать значение операнда А, в противном случае в блоке*

GENERATE может быть получен отрицательный интервал времени, который вызовет остановку по ошибке «Отрицательное время задержки».

Рассмотрим три дополнительных операнда: С – смещение интервалов, D – ограничитель, E – уровень приоритета.

Смещение интервалов (первоначальная задержка) С – это момент времени, в который в блоке GENERATE должен появиться первый транзакт. После этого первого прихода все остальные приходы транзактов возникают в соответствии с распределением интервалов времени, задаваемых операндами А и В. Операнд С можно использовать как для ускорения, так и для замедления прихода первого транзакта или для указания прихода в нужный момент времени. Начальная задержка может быть меньше, равна или больше среднего времени, заданного операндом А. Когда операнд С не используется, интервалы генерирования определяются операндами А и В (они не оказывают влияния на задержку). Операнд С может быть таким же, как и операнды А и В.

Операнд D задаёт граничное значение общего числа транзактов, которые могут войти в модель через данный блок GENERATE в течение времени моделирования. Когда это число достигнуто, данный блок GENERATE перестаёт быть активным. Если не определено граничное значение (операнд D не используется), блок GENERATE остаётся активным в течение всего времени моделирования, т. е. по умолчанию ограничения на количество создаваемых транзактов нет.

Операнд E устанавливает класс приоритета каждого из транзактов, входящих в модель через данный блок GENERATE. Для задания приоритетов с целью повышения эффективности работы GPSS World рекомендуется использовать последовательность целых чисел 0, 1, 2, 3, . . . Чем выше число, тем выше приоритет. Если операнд E не используется, по умолчанию приоритет генерируемых данным блоком GENERATE транзактов равен нулю.

Операнды D и E могут задаваться так же, как и операнды А, В и С, но при этом могут принимать значения только целых положительных и целых чисел соответственно.

В любом блоке GENERATE должен быть *обязательно* задан либо операнд А, либо операнд D. *Нельзя использовать в качестве операнда параметры транзакта.* Необходимо также помнить, что транзакт не должен входить в блок GENERATE. Если транзакт пытается это сделать, возникает ошибка выполнения. В модели блоку GENERATE должны предшествовать команды определения EQU, INITIAL, FUNCTION, VARIABLE,

BVARIABLE, FVARIABLE. Это делается для того, чтобы СЧА в блоке GENERATE, который ссылается на них, давали нужные результаты.

2. START, TERMINATE

Команда START используется для запуска процесса моделирования. Она имеет следующий формат записи:

START А, [B], C, [D]

Операнд А задаёт значение *счётчика завершения*, определяющего момент окончания прогона модели. Может быть только целым положительным числом.

Операнд В – операнд вывода статистики. Он может быть NP («нет вывода данных») либо опущен. При задании NP стандартный отчёт не выводится. По умолчанию выводится стандартный отчёт.

Операнд С не используется и сохранён для совместимости с описаниями более ранних версий GPSS. Он использовался как счётчик «кадров» для периодического вывода отчёта. Эта функция доступна при использовании в модели более одной команды START и REPORT.

Операнд D определяет необходимость вывода содержимого списков событий. Если его указать любым положительным целым числом, например, 1, списки текущих и будущих событий включаются в стандартный отчёт и выводятся. Если операнд D не указан, по

умолчанию списки в стандартном отчёте не выводятся.

Команду START можно сразу указывать в конце программы модели при её подготовке. Тогда после трансляции модели, т. е. создания объекта «процесс моделирования», начинается моделирование. Эту же команду можно вводить в программу в интерактивном режиме. Однако может возникнуть необходимость завершить моделирование не по истечении какого – либо времени, а после обработки определённого количества транзактов, имитирующих, например, изготовленные детали, переданные по каналу связи сообщения и т. д. В этом случае сегмент задания времени моделирования не нужен. Для организации такого способа завершения моделирования необходимо сделать следующее: в блоках TERMINATE, которые выводят из модели транзакты, указать число, на которое уменьшается счётчик завершения моделирования. В команде START указать то число, деление которого на указанное в блоке TERMINATE число даст требуемое количество транзактов. Например, требуется завершить моделирование после изготовления 100 деталей:

```
.....  
terminate      1  
.....  
terminate      1  
.....  
terminate  
start          100
```

Блоков TERMINATE, которые выводят из модели транзакты, соответствующие изготовленным деталям, может быть несколько. Все эти блоки должны иметь 1 в качестве операнда A. У остальных блоков TERMINATE, если они есть в модели, операнд A должен отсутствовать.

В модели может быть любое число блоков TERMINATE. Блок имеет следующий формат записи:

```
terminate      [A]
```

Значением операнда A является число единиц, на которое блок уменьшает значение счётчика завершения. Может быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*<параметр>. По умолчанию операнд A равен нулю. В этом случае транзакт уничтожается, а значение счётчика завершения остаётся неизменным.

Счётчик завершения представляет собой ячейку памяти, которая хранит положительное целое число, записанное в начале процесса моделирования командой START. В процессе моделирования транзакты попадают в блок TERMINATE и в соответствии со значением операнда A вычитают определённое число из счётчика завершения. При достижении содержимым счётчика нуля моделирование завершается. В модели может быть много блоков TERMINATE, но счётчик завершения может быть *только один*.

Когда пользователь подготавливает модель, он задаёт время моделирования, указывая в команде START значение счётчика завершения. Поскольку пути транзактов могут иметь различные пути следования в модели, каждый блок TERMINATE может либо уменьшать, либо не уменьшать значение счётчика моделирования.

3. TRANSFER

Предназначен для передачи входящего в него транзакта в другой блок модели. Имеет следующий формат:

```
transfer      [A],[B],[C],[D]
```

Все режимы блока TRANSFER, кроме безусловного, *выборочные*, т. е. отличаются друг от друга способом выбора очередного блока, к которому должен быть направлен активный транзакт. Операнд A задаёт этот режим выбора. Существуют следующие девять

режимов работы блока TRANSFER:

- , (по умолчанию) – безусловный;
- . – статистический, выбор случайным образом одного из двух блоков;
- BOTH – последовательный выбор одного из двух блоков;
- ALL – последовательный выбор одного из нескольких блоков;
- PICK – выбор случайным образом одного из нескольких блоков;
- FN – функциональный;
- P – параметрический;
- SBR – подпрограммный;
- SIM – одновременный.

Операнд А может принимать указанные выше значения, а также может быть именем, положительным целым числом, выражением в скобках, СЧА, СЧА*параметр.

Операнды В и С задают возможные значения номеров следующих блоков или их положение. Они могут быть такими же, как и операнд А. Использование этих значений будет рассмотрено ниже при описании указанных выше режимов работы. Если операнд В отсутствует, то планировщик записывает вместо него номер блока, следующего за блоком TRANSFER.

- **Режим безусловной передачи.**

Операнд А указывается символом «,». Операнд В указывает имя блока, в который транзакт должен попытаться войти. Блок TRANSFER не может отказать транзакту во входе. Если блок, указанный в операнде В, отказывает транзакту во входе, транзакт остаётся в блоке TRANSFER.

- **Режим статистической передачи.**

Операнд А указывается символом «.» и не является зарезервированным словом, блок TRANSFER работает в режиме передачи транзакта в один из двух блоков случайным образом. Значение операнда А, записываемое после точки, рассматривается как трёхзначное число, показывающее (в долях от тысячи), какая доля входящих в блок транзактов должна быть направлена в блок С. Остальные транзакты направляются в блок В или к следующему по номеру блоку, если операнд В отсутствует.

Числовое значение операнда А может быть задано любым СЧА. При этом возможны следующие случаи:

- $A \leq 0$;
- $A \geq 1000$;
- $0 < A < 1000$.

В первом случае будет производиться безусловная передача транзакта к блоку В. Во втором случае будет осуществляться безусловная передача транзакта к блоку С. В третьем случае блок работает в описанном в этом пункте режиме.

- **Режим BOTH.**

Если в качестве операнда А используется ключевое слово BOTH, то в этом режиме каждый транзакт, вошедший в блок TRANSFER, проверяет два пути. Сначала проверяется возможность войти в блок В. Если транзакт не может войти в этот блок, он пытается войти в блок С. Если транзакт не может войти и в этот блок, он задерживается в блоке TRANSFER. Опрос блоков В и С повторяется при каждом изменении модельного времени до тех пор, пока транзакт не получит возможность войти в какой – либо блок. Если не используется операнд В, то проверяется возможность войти в следующий блок. В блоке TRANSFER операнды В и С могут быть и не метками, а номерами блоков.

- **Режим ALL.**

В этом режиме вошедший в блок TRANSFER транзакт проверяет возможность войти

в любой блок, начиная с блока, указанного операндом В, и заканчивая блоком, указанным операндом С. Операнд D определяет шаг d изменения номера проверяемого блока. Значение шага даёт возможность опрашивать определённые блоки, расположенные между теми, которые заданы операндами В и С. Далее транзакт, вошедший в блок TRANSFER, пытается войти в блок В. Если блок В занят, транзакт пытается войти в блоки с номерами $v+d$, $v+2d$, ..., z , где v - номер блока, указанного операндом В, z – номер блока, указанного операндом С. При этом $z=v+nd$, где n – некоторое целое положительное число. Если не используется операнд С, проверяется только один блок. Поскольку обычно в качестве операндов В и С записываются метки блоков, то блоки следует располагать таким образом, чтобы при присвоении номеров разность между номерами блоков, указанных в операндах В и С, была кратна шагу, указанному в операнде D. Также операнды В и С могут быть и не метками, а номерами блоков.

- **Режим PICK.**

В этом режиме из последовательности блоков с номерами $v+1$, $v+2$, ..., n , v - номер блока, указанного операндом В, n – номер блока, указанного операндом С, случайным образом выбирается номер одного блока, к которому должен быть направлен активный транзакт. Все блоки, включая и указанные операндами В и С, выбираются с равной вероятностью $\frac{1}{[(n-v)+1]}$. Транзакт пытается войти только в выбранный для него блок. Если транзакт не может сразу перейти к следующему блоку, то он будет ждать в блоке TRANSFER до тех пор, пока не будет снято блокирующее условие. Номер блока С должен быть больше или равен $v+1$.

- **Режим FN.**

В этом режиме при входе транзакта в блок TRANSFER вычисляется значение функции, имя которой задано операндом В. Если результат нецелый, от него берётся дробная часть. Для определения номера следующего блока вычисленное значение складывается с операндом С (который может быть равен нулю). Если блок с вычисленным номером занят, транзакт остаётся в блоке TRANSFER до тех пор, пока не сможет перейти именно к этому блоку.

- **Режим P.**

В параметрическом режиме активный транзакт направляется к блоку, номер которого определяется как сумма значения параметра и значения операнда С. Операнд С может и не использоваться.

- **Режим SBR.**

Активный транзакт всегда направляется к блоку, номер которого указан операндом В. Номер блока TRANSFER помещается в параметр, указанный операндом С. Для возвращения из подпрограммы используется блок TRANSFER в параметрическом режиме.

- **Режим SIM.**

Введён для случая, когда требуется одновременное выполнение нескольких условий. Каждый транзакт имеет свой индикатор задержки SIM. В этом индикаторе записывается результат любой попытки транзакта войти в следующий блок. Если планировщик обнаруживает условия, препятствующие входу транзакта в блок, индикатор SIM этого транзакта устанавливается в 1. Если все условия перехода к следующему блоку удовлетворяются, индикатор остаётся равным нулю. Если не выполняется хотя бы одно из условий, индикатор устанавливается в 1. В таких случаях операнд С указывает на блок, в котором выполнялось первое условие, и транзакт выполняет проверки всех условий, пока они не будут выполняться одновременно. Блок ADVANCE также устанавливает индикатор в 0. Проверка состояний связана с блоками, которые могут задерживать транзакты. Для проверки условий используется блок GATE. Данный режим используется редко, так как более эффективным является управление большого количества объектов с помощью

булевых переменных и блока TEST.

4. LOOP

Предназначен для организации циклов в модели. Имеет следующий формат:

loop A, B

Операнд A – параметр транзакта или параметр цикла, в котором содержится число повторений какого – либо участка модели.

Операнд B – метка блока, с которого начинается цикл.

Операнды A и B могут быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*параметр.

Функции, выполняемые блоком LOOP, могут быть реализованы, следующим образом:

```
.....  
assign 2,1          ;в параметр 2 активного транзакта заносится 1  
met1 .....         ;начало цикла  
.....             ;тело цикла  
assign 2+,1         ;инкремент на 1 значения параметра 2  
test le P2,10,met1  ;проверка значения параметра 2 на равенство 10, при  
                   ;НЕВЫПОЛНЕНИИ условия цикл продолжается
```

5. DISPLACE

Предназначен для нахождения любого транзакта и перемещения его к новому блоку.

Формат блока:

displace A, B, [C], [D]

Операнд A – номер транзакта, который необходимо переместить.

Операнд B – метка, к которой перемещается транзакт, указанный операндом A.

Операнд C – номер параметра перемещаемого транзакта, в который записывается оставшееся до конца его обслуживания время, если он находился в списке будущих событий.

Операнд D – метка альтернативного блока для активного транзакта. Если операнд отсутствует, активный транзакт переходит к следующему блоку.

Операнды A ,B, C и D могут быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*параметр.

Может быть 3 случая отысканного блока:

- транзакт есть в модели и не находится в списке будущих событий:
- в этом случае транзакт перемещается к блоку, указанному операндом B
- транзакт есть в модели и находится в списке будущих событий:

в этом случае определяется время, оставшееся до его повторного ввода в процесс моделирования, и записывается в параметр, указанный операндом C, причём, если параметр с таким именем не существует, то он создаётся, а транзакт перемещается к блоку, указанному операндом B

- транзакта с нужным номером в модели нет:
- в этом случае активный транзакт пользуется операндом D.
- Когда транзакт перемещается к новому блоку, он исключается из списков:
 - будущих событий;
 - отложенных прерываний (для прерывающих транзактов);
 - задержки (в порядке приоритета);
 - пользователя;
 - повторных попыток.

Транзакт в том же случае не исключается из списков:

- текущих событий;

- прерываний;
- групп.

При перемещении прерванные выполнения в устройствах не сбрасываются. Это означает, что перемещаемый транзакт продолжает занимать устройство.

6. RMULT

Обращаются к генератору без предварительного его объявления посредством СЧА RN_n, где n – номер генератора. Начальное число генератора случайных чисел совпадает с номером n генератора, причём генератор запускается с указанного числа.

При моделировании может возникнуть необходимость исследовать:

- различные варианты модели при одном и том же входном потоке случайных чисел;
- один или несколько вариантов модели при различных потоках случайных чисел.

Первая необходимость легко реализуется, поскольку генераторы позволяют воспроизводить одни и те же последовательности равномерно распределённых случайных чисел.

Во втором случае GPSS предоставляет возможность менять начальные числа первых семи RN1...RN7 генераторов, а, следовательно, и формировать различные последовательности. Для этого в GPSS имеется команда RMULT следующего формата:

rmult [A], [B], [C], [D], [E], [F], [G]

Операнд A – начальное число для RN1, B – для RN2, C – для RN3, D – для RN4, E – для RN5, F – для RN6, G – для RN7.

rmult , 112, 17, , , 295

Устанавливаются начальные числа генераторов 2, 3, 6. Остальные значения остаются без изменений.

СЧА RN принимает целые значения от 0 до 999 включительно, а при вычислении случайных функций используется случайное число, выбираемое от 0 до 0,999999 включительно.

7. ADVANCE

Используется для задержки транзакта в течение некоторого интервала модельного времени. Чаще всего этот интервал задаётся случайной переменной. Формат записи блока:

advance A, [B]

Операнд A задаёт среднее время обслуживания, операнд B – способ модификации операнда A. Каждый из операндов может быть именем, числом, выражением в скобках, СЧА или СЧА*параметр.

Как и в блоке generate, здесь также операнды могут быть двух типов: модификатор – интервал и модификатор – функция. При использовании сохраняемых ячеек они должны быть предварительно инициализированы при помощи блока initial. Также блок advance может использовать ссылки на параметры транзактов (предварительно инициализированные).

Данный блок никогда не препятствует входу транзакта. Любое число транзактов может находиться в этом блоке одновременно. Время задержки может быть также равным 0.

advance P1,FN\$Exper

advance 56.7,23.2

advance (Exponential(7,0,MX\$Sred(P2,P\$Stolb)))

Решение типовых задач

Пусть приходящие с периодом [10±5] секунд заявки обслуживаются тремя устройствами, распределение между которыми происходит по дискретно – равномерному распределению. В каждом буфере транзакт задерживается на 5 секунд. Номер очередного

устройства записывается в параметр активного транзакта, причём транзакты уничтожаются, если выбранное устройство занято. Необходимо подсчитать количество уничтоженных транзактов и вероятность такого уничтожения (если таких данных в отчёте нет, рекомендуется увеличить время задержки в каком – то одном или во всех трёх устройствах). Подсчитать количество обслуженных каждым из буферов заявок. В модель ввести 100 транзактов.

	ter1	bvariable	f\$b1 ;ter1-булева переменная
	ter2	bvariable	f\$b2
	ter3	bvariable	f\$b3
	ent	fvariable	0 ;ent-арифметическая переменная, равная 0,
	term	fvariable	0
	serv1	fvariable	0
	serv2	fvariable	0
	serv3	fvariable	0
	potk	fvariable	0
	val	matrix	,100,2 ;матрица 100 строк, 2 столбца
		generate	10,5,,100
		rmult	1
		assign	num,(duniform(rn1,1,3)) ;присвоить значение
параметру num		msavevalue	val+,xn1,1,*num
		msavevalue	val+,xn1,2,xn1
		savevalue	ent+,1 ;к текущему значению переменной ent
прибавляется 1			
		transfer	,met1
	met1	test e	*num,1,met2
		test e	bv\$ter1,0,met5
		seize	b1 ;занять обработчик/устройство/ОКУ b1
		advance	5 ;задержка транзакта на 5 единиц модельного
времени			
		release	b1 ;освободить обработчик/устройство/ОКУ b1
		savevalue	serv1+,1 ;число обслуженных первым
устройством			
		transfer	,met3
	met2	test e	*num,2,met4
		test e	bv\$ter2,0,met5
		seize	b2
		advance	5
		release	b2
		savevalue	serv2+,1 ;число обслуженных вторым
устройством			
		transfer	,met3
	met4	test e	*num,3,met3
		test e	bv\$ter3,0,met5
		seize	b3
		advance	5
		release	b3
		savevalue	serv3+,1 ;число обслуженных третьим
устройством			
		transfer	,met3

met5	savevalue	term+,1	;подсчёт числа отказанных в обслуживании
	savevalue potk,(x\$term/x\$ent)		;подсчёт вероятности отказа
	transfer	,met3	
met3	terminate	1	
	start	100	

Для полного понимания данного примера к нему необходимо будет вернуться после успешного изучения л/р №4.

Задачи для самостоятельного решения {

Примечание

Задания 1, 2 ОБЯЗАТЕЛЬНО должны быть выполнены после изучения п/з №4. На данном этапе обязательными к выполнению являются задания 3, 4 и 5.

}

Здесь и далее начало и конец примечания указываются знаками { и } соответственно.

1) Пусть приходящие в интервале $[5 \pm 2.5]$ минуты заявки случайным образом распределяются между тремя блоками – обработчиками, один из которых искажает информацию настолько, что из него не выходит ни одна поступившая ему заявка. Время задержки в других блоках равно 1 минуте. Подсчитайте количество обработанных каждым блоком заявок, а затем процент искажённых. По количеству обработанных каждым блоком заявок сделайте вывод о наиболее загруженном блоке.

Замечание: необходимо смоделировать процесс в течение 8 часов модельного времени. Время обработки и появления каждого блока необходимо указывать в секундах.

2) Пусть некоторое передающее устройство с интервалом в 2,8 минуты передаёт последовательно каждому из трёх блоков – приёмников сообщение. В связи с несовершенством линии передачи каждое пятое сообщение признаётся промежуточным устройством - ретранслятором повреждённым и маркируется таким образом, чтобы блоки – приёмники уничтожили его. Блоки – приёмники тратят на обработку любого пришедшего сообщения 1 секунду модельного времени. Подсчитать процент неповреждённых сообщений. Время моделирования – 2 часа. Время поступления и обработки сообщений указывается в минутах.

Замечание: для уяснения значений блоков и применённых системных числовых атрибутов при необходимости обращаться к Приложениям №№ 1 – 4.

1) Заявки поступают в СМО с интервалом в 7 минут, обработка каждой из них составляет 5 минут. Провести обработку 100 заявок.

2) Заявки поступают в СМО по дискретно – равномерному закону в интервале от 2 до 15 минут и проходят обработку на трёх параллельных каналах, в которых обработка происходит в интервале от 1 до 2,2 минут, распределённом по тому же закону. Выбор канала происходит согласно режимам pick, all и статистическому режиму – 0,4, 0,3, 0,3. Смоделировать процесс обработки 50 транзактов, время обработки в каналах (блок advance) указывать в секундах.

3) Ответьте на вопросы преподавателя по теоретической части.

Контрольные вопросы

1. Перечислите блоки, с которыми вы познакомились в данной лабораторной работе.
2. Для чего предназначен блок GENERATE.
3. Для чего предназначен блок ADVANCE
4. Для чего предназначен блок LOOP
5. Для чего предназначен блок DISPLACE.
6. Какой закон задается с помощью модификатора – интервала?
7. Опишите Режим безусловной передачи.

8. Опишите Режим статистической передачи.
9. Опишите Режим BOTH.
10. Опишите Режим ALL.
11. Опишите Режим PICK.
12. Опишите Режим FN.
13. Опишите Режим P.
14. Опишите Режим SBR.
15. Опишите Режим SIM.

Оборудование и материалы: для выполнения данного практического занятия необходим компьютер с установленной операционной системой Windows 7 и программным продуктом: GPSS

Указания по технике безопасности: к выполнению практических работ допускаются студенты, ознакомившиеся с правилами работы в лаборатории, прошедшие инструктаж безопасности.

Задания: для выполнения практической работы необходимо выполнить следующее:

1. Изучить рекомендуемую литературу.
2. Выполнить практическую работу.
3. Ответить на контрольные вопросы.
4. Оформить отчет.

Содержание отчета: отчет по практической работе должен быть выполнен в редакторе MS Word и оформлен согласно требованиям. Требования по форматированию: Шрифт TimesNewRoman, интервал – полуторный, поля левое – 3 см., правое – 1,5 см., верхнее и нижнее – 2 см. Абзацный отступ – 1,25. Текст должен быть выровнен по ширине.

Отчет должен содержать титульный лист с темой практической работы, цель работы и описанный процесс выполнения вашей работы. В конце отчета приводятся выводы о проделанной работе.

В отчет необходимо вставлять скриншоты выполненной работы и добавлять описание к ним. Каждый рисунок должен располагаться по центру страницы, иметь подпись (Рисунок 1 – Создание подсистемы) и ссылку на него в тексте.

Список использованной литературы

1. Девянин П.Н. Управление доступом и информационными потоками: учебное пособие для студентов высших учебных заведений, обучающихся по специальностям направления подготовки 090300 - "Информационная безопасность вычислительных, автоматизированных и телекоммуникационных систем" и направлению подготовки 090900 - "Информационная безопасность" / П. Н. Девянин. Москва, 2011.

2. Гулятьев А. Визуальное моделирование в среде MATLAB. –СПб: Питер, 2010. – 432 с.

3. Емельянов В.В., Майорова В.И., Разумцова Ю.В. и др. Принятие оптимальных решений в интеллектуальных имитационных системах: Учебное пособие по курсам «Методы системного анализа и синтеза» и «Моделирование технологических и производственных процессов» – М.: Изд-во МГТУ им. Н.Э. Баумана, – 2008.

4. Кобелев Н.Б. Основы имитационного моделирования сложных экономических систем: Учеб. Пособие. – М.: Дело, 2009.

5. Советов Б.Я. Моделирование систем. Практикум: Учебное пособие для вузов/ Б.Я. Советов, С.А. Яковлев. – 2-е изд., перераб. и доп. – М.: Высш. шк., 2009.

6. Советов Б.Я., Яковлев С.А. Моделирование систем: учебник для вузов. – М.: Высш. шк., 2010.

Дополнительная:

7. Вендров А.М. Проектирование программного обеспечения экономических

информационных систем. Учеб. - М.: Финансы и статистика. – 2010.

8. Девянин П.Н. Модели безопасности компьютерных систем. Учеб. пособие для студентов вузов, обучающихся по специальности 075200 "Компьютер. безопасность" и 075500 "Комплекс. обеспечение информ. безопасности автоматизир. систем" - Москва, Сер. Высшее профессиональное образование. Информационная безопасность, 2005.

9. Емельянов А.А. и др. Имитационное моделирование экономических процессов: Учебное пособие / А.А. Емельянов, Е.А. Власова, Р.В. Дума; Под ред. А.А. Емельянова. – М.: Финансы и статистика, 2009.

10. Карпов Ю. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. - СПб.: БХВ-Петербург, 2005.

11. Кельтон В.Д., Лоу А.М. Имитационное моделирование. Классика CS. 3-е изд. - СПб.: Питер; Киев: Издательская группа BNV, 2004.

12. Семененко М.Г. Введение в математическое моделирование. М.: СОЛОН-Р, 2002.

Практическая работа №2

Переменные, матрицы, статистические таблицы.

Изучение блоков equ, initial, savevalue, msavevalue, matrix, table, qtable, tabulate.

Цель работы: изучить теоретическую часть, ознакомиться с ниже описываемыми блоками; выполнить практическую часть.

Формируемые компетенции: ПК-1, ПК-2, ПК-3, ПК-7, ПК-37

Теоретическая часть

1. Переменные

Арифметическая переменная является СЧА, который определяется пользователем. В GPSS World она может быть определена командой variable или fvariable, так как действие этих команд, а, следовательно, и вычисляемые значения одинаковы. Форматы команд:

name variable A

name fvariable A

name – имя арифметической переменной, операнд A – задаваемое пользователем выражение.

Ссылкой на арифметическую переменную является либо V\$имя, либо Vj, где j – номер переменной. Символическое имя может быть заменено номером – положительным целым числом, для чего нужно использовать команду equ:

SrSkop variable X12/MX\$Vrem(3,P1)

SrSkopequ 2

Теперь на арифметическую переменную SrSkop можно ссылаться так: V2. Команды equ нужно размещать в начале модели. В команде определения переменной, также как и в команде определения матрицы, вместо имени сразу указывать номер нельзя. На этапе трансляции произойдёт остановка по ошибке «Пропущена метка оператора».

Величиной арифметической переменной является величина заданного пользователем арифметического выражения. Значение арифметической переменной может использоваться следующим образом:

- как операнд; в этом случае значение арифметической переменной может представлять: номер объекта j; j логического атрибута (блок gate); номер параметра транзакта (блоки assign, index, loop, mark, split); значение атрибута;
- как аргумент функции;
- как значение зависимой переменной атрибутивной функции;
- как аргумент таблицы;

– как операнд другой арифметической переменной или булевой переменной.

Значение арифметического выражения вычисляется, когда активный транзакт входит в блок, оператор которого среди своих операндов содержит одну или более ссылок на арифметические переменные. Вычисленные значения имеют вещественный тип. В GPSS World промежуточные результаты и значения СЧА не округляются. Округление может задать сам пользователь с использованием библиотечной функции `int (A)`.

Следует понимать различие между командами `variable` и `fvariable`: при описании арифметической переменной командой `variable` все промежуточные вычисления и конечный результат округляются до целого значения. Для действительных переменных, определяемых командой `fvariable`, до целого значения округляется только конечный результат.

Кроме того, во избежание возможных недоразумений, особенно в режиме совместимости с GPSS/PC, имя переменной в рамках модели должно быть уникальным.

Булевы переменные с использованием одного блока дают возможность принимать решения в зависимости от состояния и значения атрибутов многих объектов GPSS. Булева переменная представляет собой логическое выражение, составленное из различных СЧА, в том числе и других булевых переменных.

Булева переменная определяется командой `bvariable`. Формат команды:

name bvariable A

name – имя булевой переменной, которое так же, как и имя булевой переменной, по желанию пользователя командой `equ` может быть заменено номером.

Операнд A – логическое выражение.

В логическом выражении можно использовать арифметические и логические операторы (см. Приложение №4). Конечное значение может быть либо 0, либо 1 (целочисленное значение).

Логические операторы связаны с объектами аппаратной категории и используются для определения состояния этих объектов. Имеются следующие логические операторы:

- FVj равен 1, если устройство j доступно, иначе – 0;
- FIj равен 1, если устройство j обслуживает прерывание, иначе – 0;
- SFj равен 1, если МКУ j заполнено полностью, иначе – 0;
- SEj равен 1, если МКУ j пусто, иначе – 0;
- SVj равен 1, если МКУ j доступно, иначе – 0;

– LSj равен 1, если логический ключ j включён, иначе – 0.

Здесь под j понимается имя аппаратного объекта или номер, присваиваемый командой equ. Список всех используемых в GPSS World СЧА относительно всех типов объектов и их пояснение представлены ниже.

Также могут использоваться операторы отношения (условные операторы) и булевы операторы (логическое И, логическое ИЛИ), приведённые также в Приложении №4. Скобки следует применять только в том случае, когда они необходимы:

Con3 bvariable (V3>7)&(FN\$Rav|LS7)

Здесь булева переменная равна 1, если выполняются сразу два условия: значение арифметической переменной с номером 3 больше 7 и логический ключ номер 7 включён.

Булева переменная может также задаваться СЧА, например:

stan bvariable V\$treat

В данном случае сначала вычисляется значение арифметической переменной treat, и если оно отлично от 0, то значение булевой переменной будет равным 1, в противном случае – 0.

Для обращения к булевым переменным следует использовать СЧА bv\$name/bvj, где j – номер, присвоенный командой equ.

В заключение следует обратить внимание на следующее: при проведении эксперимента с имитационной моделью факторы, влияние которых на искомый показатель предполагается исследовать, обязательно должны быть переменными пользователя.

2. INITIAL, EQU, SAVEVALUE

Следует отметить, что команда equ предназначена для присваивания вычисленного выражения и имеет формат:

name equ X

Здесь name – имя объекта – получателя вычисленного выражения, а X – присваиваемый результат вычисления выражения.

Выше приводилось иное применение команды equ – присваивание номера имени. Следует отметить, что это происходит в том случае, когда объект с данным именем ранее уже инициализирован.

Также имена, заданные командой equ, но в модели использующиеся как метки объектов, считаются метками.

Для первоначальной инициализации используется команда initial. Следует отметить, что она используется для *сохраняемых ячеек*, то есть ячеек, начальные значения которых

могут быть заданы перед моделированием и к которым можно обратиться из любого места модели во время моделирования.

Перед началом моделирования значения всех сохраняемых ячеек устанавливаются в ноль, но по желанию пользователя отдельным сохраняемым ячейкам могут быть присвоены ненулевые значения. Значения сохраняемых ячеек изменяются только при прямом указании пользователя.

Команда initial применяется при отсутствии активного транзакта или же когда нельзя использовать значения параметров транзактов. Формат команды:

initial A,[B]

Операнд A – может быть X\$name/Xj.

Операнд B – присваиваемое первоначальное значение или unspecified. Может быть именем, числом, строкой или unspecified. Если операнд не используется, значение ячейки становится равным 1.

{

Примечание

Не будет ошибочной следующая запись:

.....			
ent	fvariable	0	;определяется как переменная
term	fvariable	0	;определяется как переменная
pt	fvariable	0	;определяется как переменная
potk	fvariable	x\$term/x\$ent	;классическая переменная
.....			
.....			
	savevalue	ent+,1	;используется как сохраняемая ячейка
.....			
.....			
	savevalue	term+,1	;используется как сохраняемая ячейка
	savevalue	pt,v\$potk	;pt используется как сохраняемая ячейка
.....			

Здесь переменная pt используется как сохраняемая ячейка с замещением находящегося в ней значения, установленного командой fvariable в 0, значением вычисленной классической (по определению согласно п.1 настоящей п/р) переменной potk. Таким образом, к переменным, используемым в качестве сохраняемых ячеек, можно

обращаться не только с помощью СЧА V\$name/Vj, передающего вычисленное значение переменной (как это происходит с переменной potk), но и с помощью СЧА X\$name/Xj, передающего текущее значение содержимого сохраняемой ячейки (как это происходит с переменными ent и term в определении переменной potk).

}

В процессе моделирования значение сохраняемой ячейки изменяется при входе транзакта в блок savevalue. Формат записи:

savevalue A, B

Операнд A – имя или номер ячейки/переменной, в которую записывается значение, указанное в операнде B. В операнде A могут использоваться следующие модификаторы:

- + в этом случае значение, указанное в операнде B, прибавляется к текущему содержимому сохраняемой ячейки/переменной, указанной в операнде A;
- - в этом случае происходит действие, противоположное действию модификатора «+»;
- если не указан ни один из модификаторов, то значение, указанное в операнде B, замещает текущее содержимое сохраняемой ячейки/переменной, указанной в операнде A.

Операнд A может быть именем, положительным целым числом, выражением в скобках, СЧА*параметр или СЧА, а операнд B, кроме этого, может быть числом или строкой.

3. MATRIX , MSAVEVALUE, TABLE, QTABLE, TABULATE

Понятие сохраняемой ячейки применяется в GPSS и к матрицам. Эти объекты дают возможность упорядочить значения, которые нужно сохранить в виде матриц. Матрица должна быть предварительно описана. Команда описания матрицы имеет следующий вид:

name matrix A, B, C, [D], [E], [F], [G]

Метка name определяет имя матрицы и должна быть именем или числом (eq, до записи команды определения).

Операнд A не используется, так как оставлен для совместимости с более старыми версиями GPSS.

Операнды B, C, D, E, F и G могут быть только целыми положительными числами.

Операндом B задаётся число строк матрицы – максимальное количество элементов в первом измерении, операндом C – число столбцов матрицы – максимальное количество элементов во втором измерении.

Операнды D, E, F и G задают максимальное количество элементов в третьем, четвёртом, пятом и шестом измерениях соответственно.

В приведённом в п/р №1 команда

```
val matrix ,100,2
```

задаёт матрицу val из 100 строк и 2 столбцов.

Матрицы имеют единственный СЧА MX, с помощью которого можно обращаться к любому элементу матрицы. В качестве значений строк и столбцов могут использоваться только имена, целые числа и параметры транзактов.

```
MX6(13,5)
```

```
MX$Spis(User,P3)
```

В первом примере определяется элемент матрицы номер 6, стоящий на пересечении 13 строки и 5 столбца. Во втором примере для определения элемента матрицы используется переменная пользователя параметр номер три транзакта.

Перед началом моделирования или выполнения команды clear on значения всех элементов матрицы устанавливаются равными нулю. Некоторым или всем элементам по желанию разработчика могут быть присвоены ненулевые значения или они могут быть переведены в неопределённое состояние командой initial.

```
initial MX5(2,6), 48
```

```
initial Gros, 48
```

```
initial Calc
```

```
initial MainResult, unspecified
```

В первом примере элементу матрицы номер 5, стоящему на пересечении 2 строки и 6 столбца, присваивается значение 48. Во втором примере всем элементам матрицы присваивается значение 48. В третьем примере, в связи с отсутствием операнда B, всем элементам матрицы присваивается значение 1. В четвёртом примере все элементы матрицы будут переведены в неопределённое состояние с помощью ключевого слова unspecified. Этим переводом матрица будет подготовлена к эксперименту, в ходе которого могут быть неполученные данные. Неопределённое состояние элемента матрицы указывает на то, что данных нет. Если же элементы матрицы были бы переведены в ноль, то это можно было бы рассматривать как получение данных эксперимента, равных нулю. После эксперимента матрица с его результатами передаётся библиотечной процедуре дисперсионного анализа (anova), которая элементы, находящиеся в состоянии unspecified, обрабатывает как пропущенные данные вместо нулевых значений.

Для записи в процессе моделирования значений в матрицы, а также для изменения

значений, записанных в матрицы, служит блок msavevalue:

msavevalue A, B, C, D

В операнде A задаётся имя или номер матрицы. Крайним правым символом операнда могут быть модификаторы, описанные в п. 2 настоящей п/р.

В операнде B задаётся номер строки, в операнде C – номер столбца матрицы.

Операнд D определяет значение, которое должно храниться, прибавляться или вычитаться.

Операнды могут быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*параметр.

msavevalue val+,xn1,1,*num

msavevalue val+,xn1,2,xn1

В первом примере в таблицу с именем val записывается текущее значение параметра num в первый столбец и в строку, указываемую СЧА XN1 – номером транзакта. Во втором примере во второй столбец таблицы и в строку, указываемую номером транзакта, записывается номер транзакта.

Для матриц существует единственный СЧА MX\$name(a,b)/MXj(a,b), указывающий текущее содержимое ячейки (a,b).

Для экспериментального получения многих параметров распределения используются *статистические таблицы* table и qtable. Команда описания таблицы table имеет следующий вид:

name table A, B, C, D

Команда определяет аргумент, а также число и ширину частотных интервалов (классов). Метка name определяет имя таблицы. Аргументом A задаётся аргумент таблицы, элемент данных, чьё частотное распределение будет табулироваться. Операнд может быть именем, выражением в скобках или СЧА.

Операндом B задаётся верхний предел первого интервала. Операнд C задаёт ширину частотного интервала – разницу между верхней и нижней границами каждого частотного класса. Операнды B и C могут быть числами или string.

Операндом D задаётся число частотных интервалов (целое положительное число).

Для сбора данных транзакт должен войти в блок tabulate с тем же именем таблицы, которое определено в блоке table. Блок tabulate помещается в ту точку модели, которая соответствует исследуемому объекту. Имеет следующий формат:

tabulate A, [B]

Операндом A задаётся имя таблицы, в которую табулируется значение аргумента.

Операндом В (**весовой коэффициент**) задаётся число единиц, которые должны быть занесены в тот частотный интервал, куда попало значение аргумента. Если операнд отсутствует, то по умолчанию он равен 1.

Операнды А и В могут быть именем, выражением в скобках, СЧА или СЧА*параметр. Кроме того, операнд А может быть только положительным числом, а операнд В - положительным числом. Например:

```
VrRem table      P$ReaLvs,8.2,5.5,10
.....
      tabulate    VrRem
```

Аргументом таблицы является СЧА P\$ReaLvs с верхним пределом первого интервала 8.2, шириной 5.5 и числом интервалов 10. Каждое значение табулируемого аргумента P\$ReaLvs, меньшее или равное 8.2, увеличивает частоту первого частотного класса таблицы на 1. Если аргумент не попадает в первый класс, то класс определяется делением на операнд С оператора table. Одновременно корректируются текущие СЧА таблицы: счётчик входов в таблицу ТС, среднее время ожидания ТВ и стандартное отклонение времени ожидания TD. собранная в таблице статистика выводится в стандартный отчёт. При использовании в операнде В блока tabulate весового коэффициента последний будет добавляться к значению частоты частотного класса. Весовой коэффициент применяется также для среднего и стандартного отклонения, что равносильно нескольким входам в блок tabulate.

Кроме таблиц table могут использоваться Q – таблицы, являющиеся средством получения распределения только времени пребывания транзакта в очереди. Формат описания Q – таблицы такой же, как и table. Отличие состоит в том, что *операндом А задаётся имя очереди*. Назначение операндов В, С и D такое же, как и в команде table. Операнд В может быть нулём или положительным числом. Для создания в модели такой таблицы необходимо её предварительно определить с помощью команды вида:

```
name      qtable      A, B, C, D
```

Необходимо помнить об отличительной особенности операнда А команды qtable. При её использовании информация заносится в таблицу *автоматически при входе транзакта в блоки queue и depart* и никаких специальных мер, то есть блока tabulate, при этом не требуется.

Решение типовых задач

Пусть на обрабатывающее устройство с одним входом по равномерному закону приходят пакеты и становятся в очередь на обслуживание. Само устройство перераспределяет пакеты на 2 выхода согласно номеру, указанному в параметре out

обрабатываемого пакета. Время обработки определяется по дискретно – равномерному закону в пределах до 50. Исследовать статистику пребывания пакетов в очереди и выходную статистику моделирования 100 транзактов. Выходную статистику собрать по времени обработки пакета, указываемую в параметре tts.

que1	qtable	serv1,10,10,10
Out	table	*tts,5,10,10
ts	fvariable	duniform(1,1,50)
term	fvariable	0
	generate	(uniform(1,1,100)),,100
	assign	tts,0
	assign	out,0
	transfer	pick,met1,met2
met1	assign	out,1
	transfer	,met4
met2	assign	out,2
	transfer	,met4
met4	queue	serv1,1
	advance	10
	seize	ser
	depart	serv1,1
	advance	v\$ts
	assign	tts,v\$ts
	release	ser
	test e	*out,1,met5
	transfer	,met6
met5	test e	*out,2,met3
	transfer	,met6
met6	queue	writ
	advance	5
	depart	writ
	tabulate	Out
	transfer	,met3
met3	savevalue	term+,1
	terminate	1
	start	100

Задачи для самостоятельного решения

Замечание: по умолчанию матрицы не выводятся в стандартный отчёт, для отображения требуется: Edit→Settings→Reports, отметить флажком Matrices.

- 1) Прокомментируйте приведённый пример.
- 2) Измените программу так, чтобы транзакты случайным образом отправлялись на одно из двух обрабатывающих устройств, а выводились в один общий канал. Программно подсчитать коэффициенты использования устройств, а также для каждого из них создать Q – таблицу на входе и 2 статистические таблицы на выходе: первую по параметру tts, а другую – по абсолютному модельному времени.
- 3) Ответьте на вопросы преподавателя по теоретической части.

Контрольные вопросы

1. Для чего предназначена команда equ?
2. Для чего предназначена команда bvariable?
3. Для чего предназначена команда variable?
4. Чем различаются команды variable и fvariable?
5. Для чего предназначена команда initial?

Оборудование и материалы: для выполнения данного практического занятия необходим компьютер с установленной операционной системой Windows 7 и программным продуктом: GPSS

Указания по технике безопасности: к выполнению практических работ допускаются студенты, ознакомившиеся с правилами работы в лаборатории, прошедшие инструктаж безопасности.

Задания: для выполнения практической работы необходимо выполнить следующее:

5. Изучить рекомендуемую литературу.
6. Выполнить практическую работу.
7. Ответить на контрольные вопросы.
8. Оформить отчет.

Содержание отчета: отчет по практической работе должен быть выполнен в редакторе MS Word и оформлен согласно требованиям. Требования по форматированию: Шрифт TimesNewRoman, интервал – полуторный, поля левое – 3 см., правое – 1,5 см., верхнее и нижнее – 2 см. Абзацный отступ – 1,25. Текст должен быть выровнен по ширине.

Отчет должен содержать титульный лист с темой практической работы, цель работы и описанный процесс выполнения вашей работы. В конце отчета приводятся выводы о проделанной работе.

В отчет необходимо вставлять скриншоты выполненной работы и добавлять описание к ним. Каждый рисунок должен располагаться по центру страницы, иметь подпись (Рисунок 1 – Создание подсистемы) и ссылку на него в тексте.

Список использованной литературы

1. Девянин П.Н. Управление доступом и информационными потоками: учебное пособие для студентов высших учебных заведений, обучающихся по специальностям направления подготовки 090300 - "Информационная безопасность вычислительных, автоматизированных и телекоммуникационных систем" и направлению подготовки 090900 - "Информационная безопасность" / П. Н. Девянин. Москва, 2011.

2. Гуляев А. Визуальное моделирование в среде MATLAB. –СПб: Питер, 2010. – 432 с.

3. Емельянов В.В., Майорова В.И., Разумцова Ю.В. и др. Принятие оптимальных решений в интеллектуальных имитационных системах: Учебное пособие по курсам «Методы системного анализа и синтеза» и «Моделирование технологических и производственных процессов» – М.: Изд-во МГТУ им. Н.Э. Баумана, – 2008.

4. Кобелев Н.Б. Основы имитационного моделирования сложных экономических систем: Учеб. Пособие. – М.: Дело, 2009.

5. Советов Б.Я. Моделирование систем. Практикум: Учебное пособие для вузов/ Б.Я. Советов, С.А. Яковлев. – 2-е изд., перераб. и доп. – М.: Высш. шк., 2009.

6. Советов Б.Я., Яковлев С.А. Моделирование систем: учебник для вузов. – М.: Высш. шк., 2010.

Дополнительная:

7. Вендров А.М. Проектирование программного обеспечения экономических информационных систем. Учеб. - М.: Финансы и статистика. – 2010.

8. Десянин П.Н. Модели безопасности компьютерных систем. Учеб. пособие для студентов вузов, обучающихся по специальности 075200 "Компьютер. безопасность" и 075500 "Комплекс. обеспечение информ. безопасности автоматизир. систем" - Москва, Сер. Высшее профессиональное образование. Информационная безопасность, 2005.

9. Емельянов А.А. и др. Имитационное моделирование экономических процессов: Учебное пособие / А.А. Емельянов, Е.А. Власова, Р.В. Дума; Под ред. А.А. Емельянова. – М.: Финансы и статистика, 2009.

10. Карпов Ю. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. - СПб.: БХВ-Петербург, 2005.

11. Кельтон В.Д., Лоу А.М. Имитационное моделирование. Классика CS. 3-е изд. - СПб.: Питер; Киев: Издательская группа BHV, 2004.

12. Семенов М.Г. Введение в математическое моделирование. М.: СОЛОН-Р, 2002.

Практическая работа №3

Изменение значений параметров транзактов.

Изучение блоков assign, plus, index, mark, priority.

Изучение блоков equ, initial, savevalue, msavevalue, matrix, table, qtable, tabulate.

Цель работы: изучить теоретическую часть, ознакомиться с нижеописываемыми блоками; выполнить практическую часть.

Формируемые компетенции: ПК-1, ПК-2, ПК-3, ПК-7, ПК-37

Теоретическая часть

Первые четыре блока используются для изменения значений параметров транзактов, пятый – для изменения значения приоритета.

1. Блок assign

Каждый транзакт может иметь любое число параметров. В момент генерации транзакта все его параметры нулевые (только те, которые используются в модели). Блок assign является основным средством для задания значений параметров транзактов:

assign A, B, [C]

Операндом A задаётся номер параметра, которому приписывается значение. Может быть именем, положительным целым числом, выражением в скобках, СЧА, СЧА*параметр и следующими за ними знаками (если они необходимы): +/-.

Операнд B определяет значение, которое следует добавить, вычесть или которым следует заменить значение в параметре, заданном операндом A. Если такой параметр не существует, он создаётся со значением, равным 0. Операнд B может быть таким же, как и операнд A, а также числом и строкой.

Операнд C задаёт номер модификатора – функции. При использовании операнда C значение операнда B умножается на значение функции – модификатора. Полученное произведение становится значением, которое изменяет значение параметра, заданного операндом A. Операнд C определяет номер функции или её имя, т. е. не нужно использовать СЧА FN или FN\$ перед ним. В случае использования этих СЧА сначала вычисляется значение указываемой одним из этих СЧА функции, полученный результат используется для определения второй функции GPSS, которая также вычисляется и её значение умножается на значение операнда B.

assign 1,754.3

assign 4+,Q5

assign 3-,5.85,7

assign Name, "Plan"

assign Tr1-, (Normal(32,Sredn,SrKvOtkl)), Expdis

Следует отметить, что имена параметров транзакта могут обозначаться как цифрами, так и буквами в верхнем и/или нижнем регистре. Для использования в качестве какого – либо операнда необходимого закона распределения (список библиотечных генераторов случайных чисел находится в Приложении №4) его необходимо заключать в круглые скобки (пример 5).

2. Блок plus

Вычисляет выражение и может записать его в параметр. Имеет следующий формат:

plus A, [B]

Операнд A – выражение. Может быть именем, числом, строкой, выражением в скобках, СЧА или СЧА*параметр.

Операнд B – номер параметра транзакта, в котором сохраняется вычисленный результат. Может быть именем, положительным целым числом, выражением в скобках, СЧА

или СЧА*параметр. Если указанный в операнде параметр не существует, он создаётся.

plus (Exponential(23,0,32.7)+48.75), Itog

3. Блок index

Имеет следующий формат:

index A, B

Операнд А – номер параметра транзакта. Может быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*параметр.

Операнд В – числовое значение, изменяющее значение параметра, указанного в операнде А. Может быть именем, числом, выражением в скобках, СЧА или СЧА*параметр. Полученный результат сразу же заносится в указанный параметр транзакта.

index Slag1, (Normal(21,Mat,SreOtk)+X\$post)

Если нет параметра, указанного операндом А, происходит остановка по ошибке «Обращение к несуществующему параметру».

4. Блок mark

Заносит в активный транзакт или его параметр значение абсолютного модельного времени. Имеет следующий формат:

mark [A]

Операнд А- номер параметра транзакта, в который записывается значение абсолютного модельного времени. Может быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*параметр.

mark

mark Vhod

Если операнд А не используется, то вошедшему в этот блок транзакту по умолчанию устанавливается время входа в систему, равное абсолютному модельному времени. В противном случае значение абсолютного модельного времени (СЧА AC1) заносится в параметр, указанный операндом А. Если этого параметра нет, то он создаётся.

Блок может использоваться для вычисления времени обработки транзакта.

5. Блок priority

Используется для изменения приоритета активного транзакта. Имеет следующий формат:

priority A, [B]

Операнд А – новое значение приоритета активного транзакта. Может быть именем, числом, строкой, выражением в скобках, СЧА или СЧА*параметр.

Операнд В – буферная опция. Помещает активный транзакт по значению приоритета в список текущих событий. Может принимать значения Null/BU.

Помимо данного блока для всякого транзакта, входящего в модель, уровень значения приоритета может быть задан операндом Е блока generate. По умолчанию для каждого транзакта значение его приоритета равно 0.

Решение типовых задач

Пусть в матрицу размерностью 10×2 записывается с помощью транзактов последовательности значений геометрического распределения (Приложение №4) с вероятностью успешных испытаний Бернулли 0,2965, каждое число последовательности сохраняется в параметр транзакта gen. Во второй столбец записывается порядковый номер каждого транзакта. Листинг выглядит следующим образом:

Gen	matrix	,10,2
	generate	250,69.3,,10,0
	assign	gen,0
	transfer	,met1
met1	assign	gen,(geometric(1,0.2965))

	msavevalue	Gen,xn1,1,*gen
	msavevalue	Gen,xn1,2,xn1
	transfer	,met3
met3	terminate	1
	start	10

Замечание: список СЧА находится в Приложении №2.

Задачи для самостоятельного решения

- 1) Измените программу так, чтобы в таблицу заносились данные через тот же параметр, но с использованием блоков plus и index.
- 2) Измените размерность матрицы Gen на 4×15. В первый столбец необходимо занести последовательность чисел, распределённую по равномерному закону в интервале от 896 до 0,896, в третий столбец – абсолютное модельное время для каждого из транзактов, в четвёртый – приоритет каждого транзакта, значение которого генерируется первым генератором, начиная с числа 16.
- 3) Ответьте на вопросы преподавателя по теоретической части.

Вопросы для обсуждения

1. Для чего предназначен блок assign?
2. Для чего предназначен блок priority?
3. Для чего предназначен блок plus?
4. Для чего предназначен блок index?
5. Для чего предназначен блок mark?

Оборудование и материалы: для выполнения данного практического занятия необходим компьютер с установленной операционной системой Windows 7 и программным продуктом: GPSS

Указания по технике безопасности: к выполнению практических работ допускаются студенты, ознакомившиеся с правилами работы в лаборатории, прошедшие инструктаж безопасности.

Задания: для выполнения практической работы необходимо выполнить следующее:

9. Изучить рекомендуемую литературу.
10. Выполнить практическую работу.
11. Ответить на контрольные вопросы.
12. Оформить отчет.

Содержание отчета: отчет по практической работе должен быть выполнен в редакторе MS Word и оформлен согласно требованиям. Требования по форматированию: Шрифт TimesNewRoman, интервал – полуторный, поля левое – 3 см., правое – 1,5 см., верхнее и нижнее – 2 см. Абзацный отступ – 1,25. Текст должен быть выровнен по ширине.

Отчет должен содержать титульный лист с темой практической работы, цель работы и описанный процесс выполнения вашей работы. В конце отчета приводятся выводы о проделанной работе.

В отчет необходимо вставлять скриншоты выполненной работы и добавлять описание к ним. Каждый рисунок должен располагаться по центру страницы, иметь подпись (Рисунок 1 – Создание подсистемы) и ссылку на него в тексте.

Список использованной литературы

1. Девянин П.Н. Управление доступом и информационными потоками: учебное пособие для студентов высших учебных заведений, обучающихся по специальностям направления подготовки 090300 - "Информационная безопасность вычислительных, автоматизированных и телекоммуникационных систем" и направлению подготовки 090900 - "Информационная безопасность" / П. Н. Девянин. Москва, 2011.
2. Гультяев А. Визуальное моделирование в среде MATLAB. –СПб: Питер, 2010. – 432

с.

3. Емельянов В.В., Майорова В.И., Разумцова Ю.В. и др. Принятие оптимальных решений в интеллектуальных имитационных системах: Учебное пособие по курсам «Методы системного анализа и синтеза» и «Моделирование технологических и производственных процессов» – М.: Изд-во МГТУ им. Н.Э. Баумана, – 2008.

4. Кобелев Н.Б. Основы имитационного моделирования сложных экономических систем: Учеб. Пособие. – М.: Дело, 2009.

5. Советов Б.Я. Моделирование систем. Практикум: Учебное пособие для вузов/ Б.Я. Советов, С.А. Яковлев. – 2-е изд., перераб. и доп. – М.: Высш. шк., 2009.

6. Советов Б.Я., Яковлев С.А. Моделирование систем: учебник для вузов. – М.: Высш. шк., 2010.

Дополнительная:

7. Вендров А.М. Проектирование программного обеспечения экономических информационных систем. Учеб. - М.: Финансы и статистика. – 2010.

8. Девянин П.Н. Модели безопасности компьютерных систем. Учеб. пособие для студентов вузов, обучающихся по специальности 075200 "Компьютер. безопасность" и 075500 "Комплекс. обеспечение информ. безопасности автоматизир. систем" - Москва, Сер. Высшее профессиональное образование. Информационная безопасность, 2005.

9. Емельянов А.А. и др. Имитационное моделирование экономических процессов: Учебное пособие / А.А. Емельянов, Е.А. Власова, Р.В. Дума; Под ред. А.А. Емельянова. – М.: Финансы и статистика, 2009.

10. Карпов Ю. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. - СПб.: БХВ-Петербург, 2005.

11. Кельтон В.Д., Лоу А.М. Имитационное моделирование. Классика CS. 3-е изд. - СПб.: Питер; Киев: Издательская группа BHV, 2004.

12. Семенов М.Г. Введение в математическое моделирование. М.: СОЛОН-Р, 2002.

Практическая работа №4

Изучение одноканальных устройств.

Изучение блоков `seize`, `release`, `gate`, `test`, `queue`, `depart`.

Переменные, матрицы, статистические таблицы.

Изучение блоков `equ`, `initial`, `savevalue`, `msavevalue`, `matrix`, `table`, `qtable`, `tabulate`.

Цель работы: изучить теоретическую часть, ознакомиться с нижеописываемыми блоками; выполнить практическую часть.

Формируемые компетенции: ПК-1, ПК-2, ПК-3, ПК-7, ПК-37

Теоретическая часть

1. Определение одноканального устройства (ОКУ). Блоки `seize`, `release`

Если в какой – то момент активности элемент, требующий обслуживания (транзакт), занимает ОКУ, то он входит (или пытается войти) в блок, описывающий это ОКУ. Данный блок должен обладать следующими свойствами:

- если ОКУ уже используют, транзакт не может войти в данный блок и должен ждать в очереди;
- если ОКУ не используют, транзакт может войти в блок, но статус ОКУ изменяется на «занято».

Описанными выше свойствами обладает блок `seize`, моделирующий занятие ОКУ. Блок `advance` моделирует обслуживание транзакта в ОКУ, которое играет роль операционного элемента, и в течение определяемого им времени препятствует движению транзакта по модели. Освобождение ОКУ осуществляется блоком `release`. Следует отметить, что блок `advance` может использоваться в любом месте модели по усмотрению её создателя, но также следует указать на устойчивую модель ОКУ, описываемую этими тремя блоками: `seize` – `advance` – `release`.

Формат блока `advance` был рассмотрен в п/з №1, блоки `seize` и `release` имеют следующий формат:

```
seize      A
release A
```

В обоих блоках операнд `A` – имя ОКУ. Может быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*параметр.

Прежде, чем пройти через блок `release`, транзакт может пройти неограниченное число блоков. Если ОКУ занято, то транзакт попадает в список задержки данного ОКУ позади транзактов с таким же приоритетом и не входит в блок `seize`. Транзакту также отказывается во входе в блок `seize`, если ОКУ находится в недоступном состоянии (п/з №5). Как уже указывалось, ОКУ может иметь имя или номер. В данном случае разрешается в обоих блоках записывать на место операнда `A` номер без предварительного его присвоения командой `equ`.

2. Проверка состояния одноканального устройства – блоки `test`, `gate`

Существуют два способа проверки состояния ОКУ:

- a. с помощью блока `gate`;
- b. с помощью блока `test` и булевой переменной.

a. Используется для определения состояния ОКУ (МКУ – п/з №6) без изменения их состояния. Имеет следующий формат:

```
gate X A, [B]
```

Блок работает в двух режимах:

- отказа во входе;
- разрешения во входе и альтернативном выходе.

В первом режиме блок не пропускает транзакты, если указанный объект не находится в требуемом состоянии, иначе транзакт входит в этот блок и затем переходит к следующему по порядку блоку. Операнд А задаёт имя или номер ОКУ.

Условие задаётся одним из следующих операторов Х, связанных с ОКУ:

- NU – ОКУ, заданное операндом А, свободно;
- U – ОКУ, заданное операндом А, занято.

Операнд В содержит номер следующего блока для входящего транзакта, когда условный оператор имеет значение «ложь». Операнды А и В могут быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*параметр. Если операнд В не используется, то проверка происходит в режиме отказа. Если результат проверки не «истина», то транзакт помещается в *список повторных попыток* проверяемого объекта. Если состояние любого из объектов меняется, заблокированный транзакт снова активизируется, повторяется проверка заданного блоком gate условия. Если оно выполняется, транзакту разрешается войти в блок gate и далее перейти к следующему по порядку блоку.

```
gate u           Zrk
gate nu   Bat, Oper
```

б. Блок test описывает условие, которое проверяет при входе в него транзакт, и определяет номер следующего блока для вошедшего транзакта в зависимости от того, выполняется ли требуемое условие или нет. Так же, как и блок gate, может работать в двух режимах:

- отказа во входе;
- разрешения во входе и альтернативном выходе.

Имеет следующий формат:

```
test X           A, B, [C]
```

Операнды А и В являются сравниваемыми величинами. Они могут быть именем, числом, строкой, выражением в скобках, СЧА или СЧА*параметр. Условный оператор Х может быть выражен одним из 6 следующих операторов (Приложение №4, таблица 2):

- L;
- LE;
- E;
- NE;
- G;
- GE.

Если операнд С не указан, блок работает в первом режиме. Он помещает не вошедшие транзакты в *список повторных попыток* всех объектов, участвующих в проверке условия. Когда состояние любого из этих объектов изменяется, транзакт из списка повторных попыток активизируется и проверка производится вновь. Если условие выполняется, транзакту разрешается войти в блок test.

В булевой переменной могут использоваться следующие СЧА (Приложение №2):

- F;
- FC;
- FR;
- FT.

```
prov           bvariable   F$Server
.....
test e         BV$prov,0, PotZap
  seize           Server
  advance   P$Obrab
```

release Server

Блок test можно использовать и для других проверок и в зависимости от их результатов изменять маршруты движения транзактов.

test GE Q\$dlina, nakop, pered

Следует отметить серьёзный недостаток первого режима работы обоих блоков: данный режим (при отсутствии блока В или С соответственно) приводит к увеличению машинного времени за счёт проведения проверок для заблокированных транзактов. Уменьшение числа проверок и исключение из проверяемых тех транзактов, для которых результат проверки вряд ли будет истинным, может быть организовано с помощью блоков link и unlink (п/з №5).

3. Регистратор очереди – блоки queue, depart

Объекты типа «очередь» вводятся в GPSS для сбора статистических данных. Эта статистика должна отвечать на следующие вопросы:

- 1) сколько раз транзакты приходили в очередь;
- 2) сколько пришедших транзактов фактически присоединились к очереди и сколько их сразу без задержки заняло ОКУ;
- 3) каково было максимальное значение длины очереди;
- 4) каково было среднее число ожидающих транзактов в очереди;
- 5) каково было среднее время ожидания тех транзактов, которым пришлось ждать.

Регистраторы очередей различают заданием имён. Условия задания имён те же, что и у устройств. Разработчик вносит регистратор очереди в модель с помощью пары взаимодополняющих блоков queue (стать в очередь) и depart (покинуть очередь). Форматы записи блоков:

queue	A, [B]
depart	A, [B]

Операнд А задаёт номер или имя очереди, длину которой необходимо увеличить / уменьшить.

Операнд В задаёт число единиц, на которое увеличивается / уменьшается длина очереди. Если операнд не используется, по умолчанию В равен 1.

Оба операнда могут быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*параметр.

queue	RemQ
queue	P14, P1 ;номер занимаемой очереди указан в параметре 14
depart	RemQ
depart	V3, (V4+3.7) ;V3, V4 – арифметические переменные

Очереди, как и ОКУ, разрешается записывать вместо операнда А номер без предварительного присвоения его имени командой equ.

Длина очереди не может быть отрицательной.

. Решение типовых задач

пусть на два ОКУ подаются заявки, время обслуживания каждой из которых определяется числом, записанным в параметре serve, распределённым по экспоненциальному закону. Каждая заявка стаёт в очередь к ОКУ, причём при коэффициентах использования выше 0,8 и 0,65 соответственно ОКУ блокируются. В переменную term записать число отклонённых заявок. Листинг будет выглядеть следующим образом:

fl3	bvariable	f\$cpu1&f\$cpu2
fl1	bvariable	f\$cpu1 (fr\$cpu1>800)
fl2	bvariable	f\$cpu2 (fr\$cpu2>650)
l1	fvariable	qa\$qcpu1
l2	fvariable	qa\$qcpu2

l3	fvariable	qa\$qterm
	term	0
	generate	56,25,,300,0
	assign	serve,0
met1	transfer	,met1
	assign	serve,(exponential(359,89,26))
	test e	bv\$fl3,0,met5
	transfer	,met2
met2	queue	qcpu1
	advance	15
	depart	qcpu1
	test e	bv\$fl1,0,met4
met4	seize	cpu1
	advance	*serve
	release	cpu1
	transfer	,met3
met5	queue	qcpu2
	advance	15
	depart	qcpu2
	test e	bv\$fl2,0,met5
met3	seize	cpu2
	advance	*serve
	release	cpu2
	transfer	,met3
met3	queue	qterm
	advance	20
	savevalue	term+,1
	depart	qterm
met3	savevalue	qu1,v\$l1
	savevalue	qu2,v\$l2
	savevalue	qu3,v\$l3
	transfer	,met3
met3	terminate	1
	start	300

Задачи для самостоятельного решения

- 1) Дайте определение ОКУ. Расскажите, как задаются ОКУ;
- 2) Расскажите про возможные способы проверки состояния ОКУ. Перепишите программу с использованием блока gate;
- 3) Расскажите про регистраторы очередей;
- 4) В информационно – вычислительный центр приходят со своими задачами пользователи через 10 ± 2 мин. Если все три терминала заняты, пользователю отказывают в обслуживании. Терминалы имеют разную производительность и могут обеспечить обслуживание средней программы пользователя за 20 ± 5 , 40 ± 10 и 40 ± 20 минут соответственно. Пользователи стремятся занять терминал с наибольшей производительностью. Полученные программы поступают на 2 ЭВМ: с первого терминала на первую ЭВМ, со второго и третьего – на вторую ЭВМ. Время обработки программ на первой и второй ЭВМ равны 15 и 20 минут соответственно. Смоделировать процесс обработки 300 заданий. Определить вероятность отказа в обслуживании пользователей ИВЦ;
- 5) Предъявите тексты программ с комментариями из пунктов 1, 2 п/з №1, если они не были сделаны ранее.

Вопросы для обсуждения

6. Какими свойствами обладает блок seize?
7. Дайте определение одноканального устройства.
8. Опишите блок gate.
9. Опишите блоки queue, depart.
10. Для чего нужен список повторных попыток?

Оборудование и материалы: для выполнения данного практического занятия необходим компьютер с установленной операционной системой Windows 7 и программным продуктом: GPSS

Указания по технике безопасности: к выполнению практических работ допускаются студенты, ознакомившиеся с правилами работы в лаборатории, прошедшие инструктаж безопасности.

Задания: для выполнения практической работы необходимо выполнить следующее:

13. Изучить рекомендуемую литературу.
14. Выполнить практическую работу.
15. Ответить на контрольные вопросы.
16. Оформить отчет.

Содержание отчета: отчет по практической работе должен быть выполнен в редакторе MS Word и оформлен согласно требованиям. Требования по форматированию: Шрифт TimesNewRoman, интервал – полуторный, поля левое – 3 см., правое – 1,5 см., верхнее и нижнее – 2 см. Абзацный отступ – 1,25. Текст должен быть выровнен по ширине.

Отчет должен содержать титульный лист с темой практической работы, цель работы и описанный процесс выполнения вашей работы. В конце отчета приводятся выводы о проделанной работе.

В отчет необходимо вставлять скриншоты выполненной работы и добавлять описание к ним. Каждый рисунок должен располагаться по центру страницы, иметь подпись (Рисунок 1 – Создание подсистемы) и ссылку на него в тексте.

Список использованной литературы

1. Девянин П.Н. Управление доступом и информационными потоками: учебное пособие для студентов высших учебных заведений, обучающихся по специальностям направления подготовки 090300 - "Информационная безопасность вычислительных, автоматизированных и телекоммуникационных систем" и направлению подготовки 090900 - "Информационная безопасность" / П. Н. Девянин. Москва, 2011.

2. Гулятьев А. Визуальное моделирование в среде MATLAB. –СПб: Питер, 2010. – 432 с.

3. Емельянов В.В., Майорова В.И., Разумцова Ю.В. и др. Принятие оптимальных решений в интеллектуальных имитационных системах: Учебное пособие по курсам «Методы системного анализа и синтеза» и «Моделирование технологических и производственных процессов» – М.: Изд-во МГТУ им. Н.Э. Баумана, – 2008.

4. Кобелев Н.Б. Основы имитационного моделирования сложных экономических систем: Учеб. Пособие. – М.: Дело, 2009.

5. Советов Б.Я. Моделирование систем. Практикум: Учебное пособие для вузов/ Б.Я. Советов, С.А. Яковлев. – 2-е изд., перераб. и доп. – М.: Высш. шк., 2009.

6. Советов Б.Я., Яковлев С.А. Моделирование систем: учебник для вузов. – М.: Высш. шк., 2010.

Дополнительная:

7. Вендров А.М. Проектирование программного обеспечения экономических информационных систем. Учеб. - М.: Финансы и статистика. – 2010.

8. Девянин П.Н. Модели безопасности компьютерных систем. Учеб. пособие для

студентов вузов, обучающихся по специальности 075200 "Компьютер. безопасность" и 075500 "Комплекс. обеспечение информ. безопасности автоматизир. систем" - Москва, Сер. Высшее профессиональное образование. Информационная безопасность, 2005.

9. Емельянов А.А. и др. Имитационное моделирование экономических процессов: Учебное пособие / А.А. Емельянов, Е.А. Власова, Р.В. Дума; Под ред. А.А. Емельянова. – М.: Финансы и статистика, 2009.

10. Карпов Ю. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. - СПб.: БХВ-Петербург, 2005.

11. Кельтон В.Д., Лоу А.М. Имитационное моделирование. Классика CS. 3-е изд. - СПб.: Питер; Киев: Издательская группа BHV, 2004.

12. Семенов М.Г. Введение в математическое моделирование. М.: СОЛОН-Р, 2002.

Практическая работа №5

Изучение одноканальных устройств, списков пользователя.

Изучение блоков favail, funavail, preempt, return, link, unlink.

Цель работы: изучить теоретическую часть, ознакомиться с нижеописываемыми блоками; выполнить практическую часть.

Формируемые компетенции: ПК-1, ПК-2, ПК-3, ПК-7, ПК-37

Теоретическая часть

1. Моделирование неисправностей ОКУ – блоки favail, funavail

Для моделирования неисправностей и иных ситуаций, приводящих к выводу из строя ОКУ, в GPSS World предусмотрены блоки, реализующие их доступность и недоступность. При использовании данных блоков статистика ОКУ не искажается¹.

Недоступность ОКУ реализуется блоком funavail. Имеет следующий формат:

funavail A, [B], [C], [D], [E], [F], [G], [H]

Блок делает недоступным ОКУ с именем или номером, указываемым в операнде A.

Все транзакты, обрабатываемые блоком funavail, разделяются на 3 класса:

- транзакт, занимающий ОКУ (по seize или preempt) в момент его перевода в недоступное состояние (операнды B, C, D);
- ранее прерванные транзакты, находящиеся в списке прерываний (операнды E, F);
- транзакты, находящиеся в списке отложенных прерываний и в списке задержки ОКУ (операнды G, H).

Операндом B задаются режимы обработки транзакта, занимающего ОКУ в момент перевода его в недоступное состояние:

- CO – *режим продолжения*: продолжить обработку занимающего ОКУ транзакта во время его недоступности;
- RE – *режим удаления*: удалить и направить занимающий ОКУ транзакт к блоку, метка которого должна быть задана блоком C;
- по умолчанию – *прервать обработку и поместить в список прерываний ОКУ*: после восстановления доступности этот транзакт может занять ОКУ и заново быть обслуженным.

Операнд C – метка блока, в который будет направлен в режиме удаления транзакт, занимающий ОКУ в момент перевода его в недоступное состояние.

Операнд D – номер или имя параметра транзакта, занимавшего ОКУ в момент перевода его в недоступное состояние. Если он будет удалён, т. е. исключён из списка будущих событий данного ОКУ, то в этот параметр будет записано время, оставшееся удалённому транзакту до конца обслуживания.

Операндом E задаются режимы обработки транзактов, находящихся к моменту перевода ОКУ в недоступное состояние в списке прерываний (транзакты, обслуживание которых на данном ОКУ было ранее прервано):

- CO – *режим продолжения*: продолжить работу ОКУ во время недоступности - обслуживать транзакты из списка прерываний;
- RE – *режим удаления*: удалить и направить транзакты из списка прерываний к новому блоку, метка которого должна быть указана операндом F;
- по умолчанию – *оставить ранее прерванные транзакты в*

¹ Почему?

списке прерываний ОКУ и запретить им занимать его во время недоступности.

Операнд F указывает метку блока, к которому будут направлены транзакты из списка прерываний ОКУ, вследствие чего они не могут находиться в списке будущих событий, поэтому для них не возможности занесения в параметр времени, оставшегося до окончания обслуживания. Может использоваться и тогда, когда отсутствует операнд E. В этом случае для перемещённых к новому блоку транзактов прерывание обслуживания сохраняется.

Операндом G задаются режимы обработки транзактов, находящихся к моменту перевода ОКУ в недоступное состояние в списке отложенных прерываний, т. е. ожидающих выполнения с прерыванием, и в списке задержки:

- CO – *режим продолжения*: продолжить работу ОКУ во время недоступности – обслуживать транзакты из списка отложенных прерываний и списка задержки;
- RE – *режим удаления*: удалить и направить транзакты из списка отложенных прерываний и списка задержки к новому блоку, метка которого должна быть указана операндом H;
- по умолчанию – *оставить транзакты в списке отложенных прерываний и в списке задержки ОКУ и запретить им занимать его во время недоступности.*

Операндом H указывается метка нового блока, к которому в режиме удаления направляются транзакты из списка отложенных прерываний и списка задержки. Когда операнд G не используется, нельзя использовать и операнд H.

Недоступность ОКУ продолжается до тех пор, пока транзакт, вызвавший переход в недоступное состояние, не войдёт в блок

favail A

Данный блок изменяет состояние ОКУ на доступное, т. е. восстанавливает обычный режим вхождения транзактов в ОКУ. Все транзакты, ожидающие доступного состояния ОКУ, указанного операндом A, активизируются и могут попытаться занять его.

Замечание:

1) Операнды B . . . H относятся только к транзактам указанных трёх классов. Другие транзакты, пытающиеся прервать ОКУ, уже находящееся в недоступном состоянии, в эти классы не входят и операнды B . . . H не имеют к ним никакого отношения;

2) Перевод ОКУ в недоступное состояние и разрешение продолжать работу транзактов из указанных трёх классов даёт возможность имитировать не только отказы, но и различные дисциплины обслуживания.

Проверка состояния ОКУ в режиме недоступности проводится блоком gate (п/з №4). Условие проверки задаётся одним из следующих условных операторов X:

- 1) FNV – ОКУ, заданное операндом A, недоступно;
- 2) FV – ОКУ, заданное операндом A, доступно.

Здесь также следует отметить увеличение машинного времени при отсутствии операнда B у блока gate.

```

generate      „,1
met1          advance      (exponential(32,0,287))
              funavail      cpu1,RE,ter1,,RE,ter2,RE,ter3
              advance      (normal(21,25,4))
              favail        cpu1
              transfer      ,met1

```

.....

Данная проверка также может быть проведена с помощью блока test и булевой переменной с использованием СЧА FV (1, если ОКУ доступно, иначе 0):

```

unav1      bvariable    fv$rem1
.....
      test e            v$unav1,1,met3
      ;test e          v$unav1,0,met3
      ;test ne v$unav1,1,met3
      ;test ne v$unav1,0,met3

```

Знак «;» обозначает комментарий в GPSS World.

2. Прерывание ОКУ – блоки preempt, return

Для моделирования неисправностей ОКУ можно также применять и блоки preempt и return, однако, при этом транзакты, вызывающие прерывания, учитываются в статистике как и транзакты, обслуженные при исправном функционировании ОКУ. Это некорректно, вследствие чего и искажается статистика ОКУ. Стандартная функция данных блоков – вызывание прерывания работы ОКУ.

Если на входе ОКУ образуется очередь, то выбор транзакта для его занятия после освобождения происходит:

- 1) для транзактов с равными приоритетами по принципу FIFO;
- 2) с учётом приоритета, указанного операндом E блока generate.

При этом очередной транзакт с большим приоритетом ждёт окончания обслуживания предыдущего транзакта независимо от его приоритета. Приоритет учитывается только в образующейся очереди: в ней транзакты выстраиваются в приоритетном порядке. Такой режим функционирования организуется блоками seize и release.

Однако может возникнуть необходимость смоделировать ситуацию, когда очередной транзакт должен занять ОКУ, прервав обслуживание предыдущего транзакта. Такое прерывание называется захватом ОКУ и моделируется блоком preempt. Данный блок имеет следующий формат:

```
preempt  A, [B], [C], [D], [E]
```

Операнд A – имя или номер захватываемого ОКУ. Когда ОКУ свободно, preempt работает по принципу seize. При занятом ОКУ preempt функционирует либо в *приоритетном режиме*, либо в *режиме прерывания*. Режимы определяются операндом B:

- PR – приоритетный режим;
- по умолчанию (при неиспользовании операнда B) – режим прерывания.

В приоритетном режиме прервать обслуживание предыдущего (обслуживаемого транзакта), т. е. захватить ОКУ, может только транзакт с бóльшим приоритетом. Если приоритет претендующего на занятие ОКУ транзакта равен или ниже приоритета обслуживаемого транзакта, он помещается в список задержки ОКУ последним в своём приоритете.

Операнд C – имя или номер блока, куда должен быть направлен прерванный транзакт.

Операнд D – номер параметра прерванного транзакта, в который записывается оставшееся до окончания его обслуживания время.

Операнд E при значении RE определяет режим удаления прерванного транзакта.

Операнды A, C, D и E могут быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*параметр.

Транзакт, захвативший ОКУ, освобождает его от захвата вхождением в блок
return A

Операнд A – имя или номер освобождаемого ОКУ.

Рассмотрим примеры применения блоков preempt и return в каждом из режимов:

1) прерывание в приоритетном режиме:

а) операнд C не используется, после прерывания он с автоматически сохранённым временем до конца обслуживания автоматически возвращается в список задержки ОКУ:

```
generate       5,,,3
```

```

priorityxn1
preempt      rem1, PR
advance      11
return       rem1
test e       PR,1,met2
met1         terminate      1
met2         test e         PR,2,met3
              terminate      1
met3         terminate      1
{

```

Примечание

Данный листинг будет разобран подробно с указанием процесса вычислений времени обслуживания каждого транзакта. В дальнейшем будут указаны только значения времени обслуживания с необходимыми уточнениями.

```

}
```

Блок generate генерирует только три транзакта в моменты 5, 10 и 15 модельного времени t и перестаёт быть активным. Каждому транзакту по мере входа в блок priority получают значение приоритета согласно своему номеру: 1, 2 и 3 соответственно. Первый транзакт с приоритетом 1 в $t=5$ занимает ОКУ rem1. В $t=10$ генерируется второй транзакт с приоритетом 2. Он прерывает обслуживание первого транзакта, которому осталось до завершения обслуживания $5+11-10=6$ единиц модельного времени. Первый транзакт, в виду отсутствия операнда C, помещается в список задержки rem1.

В $t=5$ генерируется третий транзакт с приоритетом 3. Он прерывает обслуживание предыдущего транзакта, так как у него более низкий приоритет, равный 2. До завершения обслуживания второму транзакту осталось $10+11-15=6$ единиц модельного времени. Он также помещается в список задержки rem1 впереди первого транзакта, у которого приоритет ниже и равен 1.

Таким образом, в соответствии с приоритетами будут обслужены третий, второй и первый транзакты. Проследить это можно, выполнив пошаговое выполнение.

При прерывании время, оставшееся до окончания обслуживания прерванного транзакта, автоматически запоминается и затем при возобновлении обслуживания используется. Это следует из суммарного времени обслуживания трёх транзактов в данном примере: $5+11+(5+6)+(5+6)=38$ единиц модельного времени. Причём обслуживание третьего транзакта завершится в $t=15+11=26$, второго в $t=26+6=32$, третьего в $t=32+6=38$ единиц модельного времени.

Из данного примера видно, что в приоритетном режиме ОКУ может прерываться сколь угодно число раз.

b) используется операнд C:

```

generate      5,,3
priorityxn1
preempt      rem1,PR,met6
met1         advance      11
              return       rem1
met2         transfer      ,(met2+xn1)
met3         terminate     1;уничтожить транзакт с приоритетом 1
met4         terminate     1;уничтожить транзакт с приоритетом 2
met5         terminate     1;уничтожить транзакт с приоритетом 3
met6         advance      10
              transfer      ,met1

```

Обслуживание транзактов завершится:

- первого в $t=15+11=26$;
- второго в $t=26+10+11=47$;
- третьего в $t=47+10+11=68$.

с) используются операнды А, В, С и Е, операндом Е указано RE:

generate 5,,,3

priorityxn1

met1 preempt rem1,PR,met2,,RE

advance 11

return rem1

terminate 1

met2 advance 10

transfer ,met1

Обслуживание транзактов завершится:

- первого в $t=15+11=26$;
- второго в $t=26+11=37$;
- третьего в $t=37+11=48$.

d) показывается, как организовывается дообслуживание прерванных транзактов

– используются операнды А, В, С, D и Е:

generate 5,,,3

assign vob,11

priorityxn1

met1 preempt rem1,pr,met2,vob,re

advance *vob

return rem1

terminate 1

met2 advance 10

transfer ,met1

Время окончания обслуживания транзактов в порядке приоритета:

- первого в $t=15+11=26$;
- второго в $t=26+6=32$;
- третьего в $t=32+6=38$.

2) прерывание в режиме захвата:

В данном режиме, если ОКУ уже используется, активный транзакт или помещается в список отложенных прерываний, или захватывает ОКУ. Прерывание обслуживания сразу, а не помещение в список, происходит тогда, когда список отложенных прерываний пуст и обслуживаемый транзакт сам не является захватчиком. Транзактам из списка отложенных прерываний представляется право занять ОКУ ранее, чем прерванным транзактам или транзактам из списка задержки ОКУ.

а) используется только операнд А:

generate 5,,,3

assign pri,(4-xn1)

priority*pri

preempt rem1

advance 11

return rem1

met1 transfer ,(met1+xn1)

met2 terminate 1

met3 terminate 1

met4 terminate 1

Блок generate генерирует 3 транзакта. Приоритет каждого следующего транзакта ниже приоритета предыдущего: 3, 2, 1 соответственно. Такая приоритетность выбирается для того, чтобы показать, что в режиме захвата приоритеты игнорируются.

В $t=5$ блок появляется первый транзакт с приоритетом 3. ОКУ свободно и его займёт первый транзакт. В $t=10$ в модели появляется второй транзакт с приоритетом 2. Несмотря на то, что его приоритет ниже, чем у обслуживаемого транзакта, он захватывает ОКУ, а первый транзакт, которому осталось до конца обслуживания 6 единиц модельного времени, помещается в список прерываний.

В $t=10+11=21$ завершится обслуживание второго транзакта. Третий транзакт, находящийся в списке отложенных прерываний, имеет преимущество перед первым транзактом и занимает ОКУ, хотя приоритет у первого транзакта выше. Обслуживание третьего транзакта заканчивается в $t=21+11=32$ единицы модельного времени.

В $t=32$ ОКУ занимает первый транзакт. Его дообслуживание завершается в $t=32+6=38$ единиц модельного времени. Таким образом, приоритеты проигнорированы.

{

Решение типовых задач Пошаговый режим реализуется следующим образом: создаётся модель без директивы start, затем компилируется (Command→Create simulation / Ctrl+Alt+S), далее Command→Step 1 / Ctrl+Alt+1. Моделирование завершается после сообщения «Исчерпан лимит транзактов. Проверьте наличие транзактов в модели»

}

b) используются операнды A и C:

```
generate      5,,3
assign        pri,(4-xn1)
priority*pri
preempt       rem1,,met6
met1          advance      11
              return       rem1
met2          transfer      ,(met2+xn1)
met3          terminate     1
met4          terminate     1
met5          terminate     1
met6          advance      10
              transfer      ,met1
```

Время окончания обслуживания транзактов:

- второй – $t=10+11=21$;
- третий – $t=21+11=32$;
- первый – $t=32+10+11=53$;

с) используются операнды A, C, D, E, E – RE, дообслуживание прерванных транзактов производится ОКУ rem2:

```
Pri matrix      ,1,3
generate      5,,3
assign        vob,11
assign        pri,(4-xn1)
msavevalue Pri,1,xn1,*pri
priority      *pri
preempt       rem1,,met2,vob,re
advance      *vob
return       rem1
met1terminate 1
met2seize     rem2
```

advance *vob
 release rem2
 transfer ,met1

Время окончания обслуживания транзактов:

- первый – $t=(5+11-10)+10=16$;
- второй – $t=16+6=22$;
- третий – $t=15+11=26$.

Проверка состояния ОКУ в приоритетном режиме может проводиться блоком gate, а также с использованием булевой переменной и блока test. Для блока gate условие проверки задаётся одним из условных операторов X:

- I – ОКУ, заданное операндом A, прервано;
- NI – ОКУ, заданное операндом A, не прервано.

Для блока test и булевой переменной имеется СЧА FI (1, если ОКУ обслуживает прерывание, иначе 0).

3. Списки пользователя и их применение – блоки link, unlink

Для экономии машинного времени заблокированные транзакты целесообразно помещать в *списки пользователя* и оставлять их там до тех пор, пока не будут выполнены условия, позволяющие дальнейшее продвижение этих транзактов. Кроме того, размещение ожидающих транзактов в списках пользователя позволяет организовывать различные дисциплины очередей, отличающихся от дисциплины FIFO, реализованной в списке текущих событий.

Список пользователя представляет собой некоторый буфер, в который могут помещаться транзакты, временно выведенные из списка текущих событий (СТС). *В отличие от списков текущих и будущих событий транзакты вводятся в список пользователя и выводятся из него не автоматически, а по решению пользователя в соответствии с логикой модели при помощи специальных блоков.*

Для ввода транзактов в список пользователя служит блок link, который может быть использован в двух режимах – условном и безусловном:

1) в безусловном режиме блок link имеет следующий формат записи:

[имя] link A, B

Операндом A задаётся имя или номер списка пользователя, в который безусловно помещается транзакт, вошедший в блок link.

Операнд B определяет, в какое место списка пользователя следует поместить вошедший транзакт. Допустимые значения:

- FIFO;
- LIFO;
- PR – транзакты упорядочиваются по убыванию приоритета;
- P – транзакты помещаются позади тех транзактов, значения соответствующего параметра которых меньше (в порядке возрастания значения транзакта);
- M1 – транзакты помещаются в порядке возрастания относительного временного пребывания в модели.

В качестве операнда B могут использоваться и другие СЧА, кроме указанных ранее СЧА транзактов: арифметическая переменная, функция, а также выражение в скобках. В этом случае выполняется вычисление указанного операндом B для активного транзакта и для всех остальных транзактов, находящихся в списке пользователя, начиная с начала очереди. После этого производится упорядочивание транзактов в списке пользователя по убыванию вычисленного транзакта.

Условия, при которых транзакт помещается в список пользователя, в безусловном

режиме проверяются средствами, предусмотренными разработчиком модели. Например, направить транзакт в список пользователя в случае занятости ОКУ можно так:

```

.....
gate nurem1, wait
seize          rem1

.....
wait          link          otst, FIFO
.....

```

Или так:

```

.....
gate nurem1, met1
link          otst, FIFO
met1 seize    rem1
.....

```

В только что приведённых примерах предполагается, что список пользователя неограничен, но при моделировании реальных систем список пользователя может использоваться для имитации, например, входного накопителя (второй вариант имитации – МКУ (п/з №6)), ёмкость которого, как правило, ограничена. Это ограничение можно реализовать следующим образом:

```

emk          equ          10
.....
gate nurem1,wait
seize          rem1

.....
wait          test l      ch$otst, emk, term1
link          otst, FIFO

.....
или
emk          equ          10
.....
gate u        rem1, met1
test l        ch$otst, emk, term1
link          otst, FIFO
met1 seize    rem1
.....

```

Для вывода одного или нескольких транзактов из списка пользователя и помещения их обратно в список текущих событий служит блок unlink, имеющий следующий формат:

[имя] unlink X A, B, C, [D], [E], [F]

Операндом А указывается номер или имя списка пользователя.

Операнд В – метка блока, в который переходят выведенные из списка пользователя транзакты.

Операндом С указывается число выводимых транзактов или ALL (выводятся все находящиеся в списке транзакты). Если не используется, подразумевается ALL.

Операнды D и E вместе с условным оператором X определяют способ и условия вывода транзактов из списка пользователя. Значения X те же, что и у блока test. В случае, когда условный оператор должен использоваться, но не указан, по умолчанию принимает значение E. Если данные операнды не используются, не указывается и условный оператор X. В этом случае транзакты выводятся с начала списка, а их количество определяется обязательным в этом случае операндом С. Операнд D может быть булевой переменной,

номером параметра транзакта или ключевым словом BACK.

Если операнд D является булевой переменной, операнды E и X не используются. Булева переменная вычисляется относительно транзакта, находящегося в списке пользователя. Если результат не нуль, т. е. условие вывода выполняется, то транзакт выводится. Количество выводимых транзактов указывается операндом C, но может быть выведено и меньше, чем указано операндом C: по числу ненулевых результатов вычисления булевой переменной. Кроме того, транзактов в списке пользователя может быть меньше, чем указано операндом C.

Если операндом D указано ключевое слово BACK, то операнды E и X также не используются, а транзакты выводятся с конца списка в количестве, указанном обязательным в этом случае операндом C.

Если операнд D не булева переменная и не ключевое слово BACK, то операнды E и X должны быть указаны. Операнд D вычисляется относительно транзакта, находящегося в списке пользователя, и используется в качестве номера параметра, значение которого сравнивается с результатом вычисления операнда E.

Если операнд D задаёт параметр, а операнд E не используется, значение параметра транзакта из списка пользователя сравнивается со значением такого же параметра выводящего транзакта. Если они равны, транзакт выводится из списка пользователя. Количество выводимых транзактов определяется операндом C.

Операндом F указывают имя блока, куда переходит транзакт, выходящий из блока unlink, если из списка не выведен ни один транзакт. Если операнд F не используется, блок переходит в следующий за unlink блок вне зависимости от выведенных транзактов.

unlink 4, apd, 1 – выводит из списка пользователя номер 4 один транзакт с начала списка и направляет его в блок с меткой apd;

unlink otst, mars, 1, back – выводит из списка пользователя с именем otst один транзакт с конца списка и направляет его в блок с меткой mars;

unlink e p\$wiw, app1, all, app2, p\$App2, app3 – выводит из списка пользователя, номер которого записан в параметре wiw выводящего транзакта, и направляет в блок с меткой app1 все транзакты, содержимое параметра App2 которых равно содержимому одноимённого параметра выводящего транзакта. Если таких транзактов не окажется, выводящий транзакт будет направлен в блок с меткой app3, иначе – к следующему за unlink блоку.

Следует отметить особенности выполнения блока unlink: во – первых, если операнды D и E содержат ссылки на СЧА транзактов, операнд D вычисляется относительно транзактов, находящихся в списке пользователя, а операнд E - относительно активного транзакта; во – вторых, после вывода транзактов из списка планировщик продолжает или начинает продвижение транзакта с наивысшим приоритетом, а при равенстве приоритетов отдаёт предпочтение выводящему транзакту.

Ниже приведены примеры различных дисциплин обслуживания, могущих быть реализованными блока link, unlink в безусловном режиме:

а) в одноканальной модели требуется организовать такую дисциплину обслуживания, при которой заявки не обслуживаются, если они находятся в очереди более чем единиц модельного времени:

vrem	equ	250
preb	bvariable	(ac1-p\$vob) 'le' vrem
	generate	(exponential(23,0,120)),,10
	assign	nb,(exponential(34,0,150))
	gate nu	pc,otst
app1	seize	pc
	advance	*nb

	release	pc
	unlink	pol,app1,1,bv\$preb,,term2
term1	terminate	
term2	terminate	
otst assign		vob,ac1
	link	pol,p\$vob
	generate	3600
	terminate	1

b) список пользователя имитирует накопитель (буфер) какой – либо реальной системы, т. е. ёмкость списка пользователя ограничена не количеством находящихся в нём транзактов, а числом, например, байт:

		10000
	generate	(exponential(12,0,12.5))
	assign	large,(normal(13,5450,350))
	gate u	pc,met1
	test le	*large,(emk-x\$emknak),met3
	savevalue	emknak+,*large
	link	otst,p\$large
met1	seize	pc
	advance	(exponential(12,0,32.5))
	release	pc
	unlink	otst,met2,1
	terminate	
met2	savevalue	emknak-,*large
	transfer	,met1
met3	terminate	1

2) в условном режиме блок link имеет следующий формат записи:

[имя] link A, B, [C]

Операндом C указывается метка блока, к которому переходит активный транзакт в случае, если индикатор (флаг) списка пользователя установлен в 0 (выключен).

Индикатор имеется у каждого списка пользователя. Когда операнд C не используется, т. е. блок link работает в безусловном режиме, индикатор устанавливается в 1(включено) и все вошедшие в блок link транзакты помещаются в список пользователя.

```

.....
link      nak,m1
met1seize can
advance  15
release  can
unlink    nak,met1,1

```

Выше приведённый код работать не будет (почему?). Если мы используем операнд C, то получим работоспособный код:

```

.....
link      nak,m1,met1
met1seize can
advance  15
release  can
unlink    nak,met1,1

```

Индикатор списка пользователя управляется блоками link и unlink. Если в блоке link

используется операнд С, блок unlink при обнаружении пустоты списка пользователя, сбрасывает его индикатор в ноль. Первый транзакт, входящий в блок link, находит индикатор выключенным и переходит на метку блока, указанную операндом С. После этого блок link переводит индикатор в 1. Если следующий транзакт войдет в блок link раньше, чем предыдущий закончит своё обслуживание в ОКУ, то попадает в список пользователя. В противном случае предыдущий транзакт проходит через блок unlink, и он, получив информацию о пустоте списка пользователя, сбрасывает индикатор в 0. Следующий транзакт направится по метке, указанной в операнде С блока link, а не будет помещён в список пользователя.

При использовании блоков link и unlink нет необходимости использовать дополнительно регистраторы очередей, так как почти все те же данные можно получить из статистики о списке пользователя.

Задачи для самостоятельного решения

- 1) Прокомментируйте все примеры из пункта 2, в которых указано только время окончания обслуживания транзактов, в том смысле, что привести полный ход вычислений этого времени по описанным примерам в пунктах 2.1.a и 2.1.b;
- 2) Пусть ОКУ выходит из строя. Используя список пользователя, необходимо удалить две трети транзактов, находящихся в списке;
- 3) Смоделируйте следующие модели обслуживания в безусловном режиме работы блока unlink:
 - в операнде В используются СЧА класса Р: упорядочить по значению времени обслуживания;
 - в операнде В указан СЧА M1;
 - в операнде В указан СЧА PR, но обслуживаются транзакты, начиная с наименьшего приоритета.
- 4) Ответьте на вопросы преподавателя по теоретической части.

Вопросы для обсуждения

1. Опишите блоки favail, funavail.
2. Для чего предназначен операнд В?
3. Для чего предназначен операнд С?
4. Для чего предназначен операнд D?
5. Для чего предназначен операнд H?

Оборудование и материалы: для выполнения данного практического занятия необходим компьютер с установленной операционной системой Windows 7 и программным продуктом: GPSS

Указания по технике безопасности: к выполнению практических работ допускаются студенты, ознакомившиеся с правилами работы в лаборатории, прошедшие инструктаж безопасности.

Задания: для выполнения практической работы необходимо выполнить следующее:

1. Изучить рекомендуемую литературу.
2. Выполнить практическую работу.
3. Ответить на контрольные вопросы.
4. Оформить отчет.

Содержание отчета: отчет по практической работе должен быть выполнен в редакторе MS Word и оформлен согласно требованиям. Требования по форматированию: Шрифт TimesNewRoman, интервал – полуторный, поля левое – 3 см., правое – 1,5 см., верхнее и нижнее – 2 см. Абзацный отступ – 1,25. Текст должен быть выровнен по ширине.

Отчет должен содержать титульный лист с темой практической работы, цель работы и описанный процесс выполнения вашей работы. В конце отчета приводятся выводы о проделанной работе.

В отчет необходимо вставлять скриншоты выполненной работы и добавлять описание к ним. Каждый рисунок должен располагаться по центру страницы, иметь подпись (Рисунок 1 – Создание подсистемы) и ссылку на него в тексте.

Список использованной литературы

1. Девянин П.Н. Управление доступом и информационными потоками: учебное пособие для студентов высших учебных заведений, обучающихся по специальностям направления подготовки 090300 - "Информационная безопасность вычислительных, автоматизированных и телекоммуникационных систем" и направлению подготовки 090900 - "Информационная безопасность" / П. Н. Девянин. Москва, 2011.
2. Гулятьев А. Визуальное моделирование в среде MATLAB. –СПб: Питер, 2010. – 432 с.
3. Емельянов В.В., Майорова В.И., Разумцова Ю.В. и др. Принятие оптимальных решений в интеллектуальных имитационных системах: Учебное пособие по курсам «Методы системного анализа и синтеза» и «Моделирование технологических и производственных процессов» – М.: Изд-во МГТУ им. Н.Э. Баумана, – 2008.
4. Кобелев Н.Б. Основы имитационного моделирования сложных экономических систем: Учеб. Пособие. – М.: Дело, 2009.
5. Советов Б.Я. Моделирование систем. Практикум: Учебное пособие для вузов/ Б.Я. Советов, С.А. Яковлев. – 2-е изд., перераб. и доп. – М.: Высш. шк., 2009.
6. Советов Б.Я., Яковлев С.А. Моделирование систем: учебник для вузов. – М.: Высш. шк., 2010.

Дополнительная:

7. Вендров А.М. Проектирование программного обеспечения экономических информационных систем. Учеб. - М.: Финансы и статистика. – 2010.
8. Девянин П.Н. Модели безопасности компьютерных систем. Учеб. пособие для студентов вузов, обучающихся по специальности 075200 "Компьютер. безопасность" и 075500 "Комплекс. обеспечение информ. безопасности автоматизир. систем" - Москва, Сер. Высшее профессиональное образование. Информационная безопасность, 2005.
9. Емельянов А.А. и др. Имитационное моделирование экономических процессов: Учебное пособие / А.А. Емельянов, Е.А. Власова, Р.В. Дума; Под ред. А.А. Емельянова. – М.: Финансы и статистика, 2009.
10. Карпов Ю. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. - СПб.: БХВ-Петербург, 2005.
11. Кельтон В.Д., Лоу А.М. Имитационное моделирование. Классика CS. 3-е изд. - СПб.: Питер; Киев: Издательская группа BHV, 2004.
12. Семененко М.Г. Введение в математическое моделирование. М.: СОЛОН-Р, 2002.

Практическая работа №6

Изучение многоканальных устройств.

Изучение блоков enter, leave, gate, test, savail, sunavail.

Цель работы: изучить теоретическую часть, ознакомиться с нижеописываемыми блоками; выполнить практическую часть.

Формируемые компетенции: ПК-1, ПК-2, ПК-3, ПК-7, ПК-37

Теоретическая часть

1. Определение многоканальных устройств (МКУ), блоки enter, leave

Два или более обслуживающих устройств могут быть промоделированы на GPSS двумя или более ОКУ, располагаемыми рядом, т. е. параллельно. Так нужно поступать, когда отдельные устройства являются разнородными, то есть характеризуются различными свойствами, например, различной интенсивностью обслуживания. Однако часто различные параллельно работающие устройства являются однородными. GPSS представляет для моделирования однородных параллельных устройств специальное средство, именуемое многоканальным устройством. МКУ может быть использовано несколькими транзактами одновременно. Ограничений на число МКУ в модели нет. Для различия им дают имена.

МКУ определяется до его использования командой storage. Формат команды:

name storage A

Name — имя МКУ. Символическому имени может быть поставлен в соответствие номер командой EQU. Это необходимо, если требуется обращаться к нескольким МКУ в блоках SELECT и COUNT. Операнд A может быть только целым положительным числом. Иные способы задания емкости вызывают ошибку. В модели можно организовать функционирование МКУ в двух режимах:

Решение типовых задач

1) занятие и освобождение МКУ:

Имитируется блоками ENTER (войти) и LEAVE (выйти). Форматы блоков:

enter A, [B]

leave A, [B]

Операнд A в обоих блоках используется для указания имени, соответствующего МКУ. Операнд B задает число устройств (элементов памяти), которое должно быть занято в блоке ENTER или освобождено в блоке LEAVE. По умолчанию операнд B = 1. При B = 0 блок считается неработоспособным.

Когда транзакт входит в блок ENTER, операнд A используется для нахождения МКУ с указанным именем. Если такого МКУ нет, происходит останов по ошибке «Обращение к несуществующей памяти». Если МКУ существует и задан операнд B, он вычисляется, округляется до целого и полученный результат используется для оценки свободной емкости. Транзакт может войти в блок ENTER, если МКУ находится в доступном состоянии и достаточно емкости для выполнения запроса. В противном случае транзакт помещается в список задержки устройства в соответствии с приоритетом.

Когда транзакт входит в блок ENTER, планировщик выполняет следующие действия:

- увеличивает на 1 счетчик входов МКУ;
- увеличивает на значение операнда B (по умолчанию на 1) текущее содержимое МКУ;
- уменьшает на значение операнда B (по умолчанию на 1) доступную емкость МКУ.

Если транзакт при входе в блок ENTER запрашивает больше устройств (элементов памяти), чем определено командой STORAGE, т. е. её операнд А меньше операнда В блока ENTER, возникает ошибка «Запрос элементов памяти превышает ее общую емкость». МКУ никогда не может быть удалено из текущей модели, даже если команда STORAGE удаляется из рабочей программы. МКУ можно переопределить, т. е. изменить емкость другой командой STORAGE с тем же самым именем. Имитация обслуживания в течение какого – то промежутка времени осуществляется также с помощью блока advance.

```

nak          storage 20
.....
          enter      nak ; enter      nak,21 – ошибка
          advance    10
          leave      nak ; leave      nak, 25 – ошибка
.....

```

2) **недоступность МКУ (моделирование неисправностей):**

Моделируется блоком sunavail. Имеет следующий формат:

```
sunavail      A
```

Операнд А – имя или номер МКУ, может быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*параметр.

```
sunavail      Batr
```

Когда транзакт входит в этот блок, МКУ Batr становится недоступным. Если при переводе в недоступное состояние в МКУ находились транзакты, т. е. текущее содержимое МКУ не равнялось нулю, то обслуживание этих транзактов продолжается, пока текущее содержимое не станет равным нулю. Транзакты, которые пытаются занять МКУ во время нахождения его в недоступном состоянии, не входят в блок ENTER и помещаются в список задержки МКУ.

Нахождение в недоступном состоянии продолжается до тех пор, пока транзакт не войдет в блок SAVAIL. Формат блока:

```
savail        A
```

Операнд А – имя или номер МКУ. Может быть таким же, как и операнд А блока sunavail.

Если в момент перевода МКУ в доступное состояние в его списке задержки были транзакты, им предоставляется возможность занять МКУ в соответствии с дисциплиной «first-fit-with-skip» (первый подходящий с пропусками). Транзакты, которым будет отказано в занятии МКУ, остаются в списке задержки.

```

kan          storage 3
          generate    ,,3
          enter       kan
          advance     10
          leave       kan
          terminate
          generate     5,,1
          sunavail     kan
          advance      6
          savail       kan
          terminate    1

```

Первый блок generate в t=0 генерирует 3 транзакта, сразу же занимающих МКУ kan и начинающих обслуживаться. В момент t=5 второй блок generate генерирует один транзакт, вызывающий выход из строя МКУ и возвращающий его через 6 единиц модельного времени. Вот результаты моделирования:

START TIME	END TIME	BLOCKS
------------	----------	--------

FACILITIES	STORAGES	0.000	11.000	10
0	1			

		NAME	VALUE
		KAN	10000.000
CURRENT	LABEL	LOC	BLOCK TYPE
COUNT	RETRY		ENTRY COUNT
0	0	1	GENERATE 3
0	0	2	ENTER 3
0	0	3	ADVANCE 3
0	0	4	LEAVE 3
0	0	5	TERMINATE 3
0	0	6	GENERATE 1
0	0	7	SUNAVAIL 1
0	0	8	ADVANCE 1
0	0	9	SAVAIL 1
0	0	10	TERMINATE 1

В представленном выше примере все транзакты закончили своё обслуживание до восстановления обслуживания. Рассмотрим пример, в котором некоторые транзакты ещё не закончили своё обслуживание до восстановления доступности МКУ:

kan	storage3
generate	3,,,3
enter	kan
advance	4
leave	kan
terminate	
generate	11,,,1
sunavail	kan
advance	1
savail	kan
advance	1 ;(*)
terminate	1

Первый блок generate с интервалом в 3 единицы модельного времени генерирует 3 транзакта. Первый в t=3 занимает первый канал МКУ и заканчивает своё обслуживание в t=7. Второй транзакт занимает второй канал МКУ в t=6 и заканчивает своё обслуживание в t=10. Третий транзакт в t=9 займёт один из двух свободных каналов МКУ. В t=11 из второго

блока generate транзакт симитирует выход МКУ из строя и восстановление в t=12. Обработка третьего транзакта продолжится и закончится в t=13(если убрать строку (*), моделирование закончится в 12 единиц модельного времени, и третий транзакт будет недообслужен). Вот результаты моделирования:

FACILITIES	STORAGES	START TIME	END TIME	BLOCKS
0	1	0.000	13.000	11

NAME	VALUE
KAN	10000.000

CURRENT	LABEL COUNT RETRY	LOC	BLOCK TYPE	ENTRY COUNT
0	0	1	GENERATE	3
0	0	2	ENTER	3
0	0	3	ADVANCE	3
0	0	4	LEAVE	3
0	0	5	TERMINATE	3
0	0	6	GENERATE	1
0	0	7	SUNAVAIL	1
0	0	8	ADVANCE	1
0	0	9	SAVAIL	1
0	0	10	ADVANCE	1
0	0	11	TERMINATE	1

Таким образом, осуществить выход из строя МКУ с помощью блоков savail и sunavail невозможно.

2. Применение к МКУ блоков gate, test

1) проверка состояния блоком gate

Состояние МКУ, как и состояние ОКУ, проверяется блоком GATE такого же формата:
gate X A, [B]

Отличие состоит в значениях условного оператора X, которые могут быть следующими:

- SE — МКУ, заданное операндом A, пусто;
- SF — МКУ, заданное операндом A, заполнено;
- SNE — МКУ, заданное операндом A, не пусто;
- SNF — МКУ, заданное операндом A, не заполнено;

- SNV — МКУ, заданное операндом А, не доступно;
- SV — МКУ, заданное операндом А, доступно.
- Блок GATE также работает в двух режимах:
 - отказа во входе;
 - разрешении во входе и альтернативном выходе.

2) проверка состояния с помощью блока test и булевой переменной

При проверке состояния МКУ блок TEST имеет тот же формат записи и те же условные операторы, что и при проверке состояния ОКУ. В булевой переменной допускается использовать следующие логические операторы, связанные с МКУ:

- SF равен 1, если МКУ заполнено полностью, иначе — 0;S
- E равен 1, если МКУ пусто, иначе — 0;
- SV равен 1, если МКУ доступно, иначе — 0,
- а также СЧА МКУ:
- S — число занятых каналов МКУ;
- SA — среднее значение занятых каналов МКУ;
- SC — счетчик использования МКУ;
- SR — коэффициент использования МКУ;SM — максимальное число занятых каналов МКУ
- ;ST — среднее время использования одного канала МКУ.

Как видно, SE, SF и SV можно использовать как в блоке GATE, так и в булевых переменных. Если же в блоке GATE в качестве условного оператора X указать, например, SA, то при трансляции будет ошибка «Недопустимое использование спецификатора класса СЧА». Если же в булевой переменной указать, например, SNV, также будет ошибка «Предполагается фактор выражения».

zap	equ	3
sist	storage	7
kont1	bvariable	sv\$sist'and'((7-s\$sist)>=zap)
kont2	bvariable	se\$sist
	generate	2,,,5
	test e	bv\$kont1,1,met1
	enter	sist,3
	advance	5
	leave	sist,3
	terminate	1
met1	terminate	1
	generate	,,,1
met2	advance	7
	sunavail	sist
	test e	bv\$kont2,1
	advance	1
	savail	sist
	transfer	,met2

Задачи для самостоятельного решения

1. На трёх параллельно работающих МКУ организовать обслуживание так, чтобы их коэффициенты использования были одинаковыми – 0,5;
2. Ответить на вопросы преподавателя по теоретической части.

Вопросы для обсуждения

1. Опишите блоки enter, leave.
2. Дайте определение МКУ. Для чего оно может использоваться?
3. Расскажите про способы проверки состояния МКУ;
4. Расскажите про моделирование неисправностей. Почему данные блоки не в состоянии моделировать неисправности?

Оборудование и материалы: для выполнения данного практического занятия необходим компьютер с установленной операционной системой Windows 7 и программным продуктом: GPSS

Указания по технике безопасности: к выполнению практических работ допускаются студенты, ознакомившиеся с правилами работы в лаборатории, прошедшие инструктаж безопасности.

Задания: для выполнения практической работы необходимо выполнить следующее:

1. Изучить рекомендуемую литературу.
2. Выполнить практическую работу.
3. Ответить на контрольные вопросы.
4. Оформить отчет.

Содержание отчета: отчет по практической работе должен быть выполнен в редакторе MS Word и оформлен согласно требованиям. Требования по форматированию: Шрифт TimesNewRoman, интервал – полуторный, поля левое – 3 см., правое – 1,5 см., верхнее и нижнее – 2 см. Абзацный отступ – 1,25. Текст должен быть выровнен по ширине.

Отчет должен содержать титульный лист с темой практической работы, цель работы и описанный процесс выполнения вашей работы. В конце отчета приводятся выводы о проделанной работе.

В отчет необходимо вставлять скриншоты выполненной работы и добавлять описание к ним. Каждый рисунок должен располагаться по центру страницы, иметь подпись (Рисунок 1 – Создание подсистемы) и ссылку на него в тексте.

Список использованной литературы

1. Девянин П.Н. Управление доступом и информационными потоками: учебное пособие для студентов высших учебных заведений, обучающихся по специальностям направления подготовки 090300 - "Информационная безопасность вычислительных, автоматизированных и телекоммуникационных систем" и направлению подготовки 090900 - "Информационная безопасность" / П. Н. Девянин. Москва, 2011.

2. Гулятьев А. Визуальное моделирование в среде MATLAB. –СПб: Питер, 2010. – 432 с.

3. Емельянов В.В., Майорова В.И., Разумцова Ю.В. и др. Принятие оптимальных решений в интеллектуальных имитационных системах: Учебное пособие по курсам «Методы системного анализа и синтеза» и «Моделирование технологических и производственных процессов» – М.: Изд-во МГТУ им. Н.Э. Баумана, – 2008.

4. Кобелев Н.Б. Основы имитационного моделирования сложных экономических систем: Учеб. Пособие. – М.: Дело, 2009.

5. Советов Б.Я. Моделирование систем. Практикум: Учебное пособие для вузов/ Б.Я. Советов, С.А. Яковлев. – 2-е изд., перераб. и доп. – М.: Высш. шк., 2009.

6. Советов Б.Я., Яковлев С.А. Моделирование систем: учебник для вузов. – М.: Высш. шк., 2010.

Дополнительная:

7. Вендров А.М. Проектирование программного обеспечения экономических информационных систем. Учеб. - М.: Финансы и статистика. – 2010.

8. Десянин П.Н. Модели безопасности компьютерных систем. Учеб. пособие для студентов вузов, обучающихся по специальности 075200 "Компьютер. безопасность" и 075500 "Комплекс. обеспечение информ. безопасности автоматизир. систем" - Москва, Сер. Высшее профессиональное образование. Информационная безопасность, 2005.

9. Емельянов А.А. и др. Имитационное моделирование экономических процессов: Учебное пособие / А.А. Емельянов, Е.А. Власова, Р.В. Дума; Под ред. А.А. Емельянова. – М.: Финансы и статистика, 2009.

10. Карпов Ю. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. - СПб.: БХВ-Петербург, 2005.

11. Кельтон В.Д., Лоу А.М. Имитационное моделирование. Классика CS. 3-е изд. - СПб.: Питер; Киев: Издательская группа BHV, 2004.

12. Семенов М.Г. Введение в математическое моделирование. М.: СОЛОН-Р, 2002.

Практическая работа № 7
Изучение переключателей.
Изучение блока logic.

Цель работы: изучить теоретическую часть, ознакомиться с нижеописываемыми блоками; выполнить практическую часть.

Формируемые компетенции: ПК-1, ПК-2, ПК-3, ПК-7, ПК-37

Теоретическая часть

1. Логический ключ – блок logic

Для моделирования такого оборудования, как переключатели, имеющие только два состояния, в GPSS используются логические ключи. Логический ключ может находиться в одном из двух состояний:

- включен (ON или 1);
- выключен (OFF или 0).

В зависимости от состояния ключа изменяется направление движения транзактов. Логический ключ моделируется блоком LOGIC. Формат блока:

logic X A

Операнд A — имя или номер логического ключа. Может быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*параметр.

Логический оператор X — состояние логического ключа устанавливается в зависимости от следующих его значений:

- S — логический ключ, заданный операндом A, включается;
- R — логический ключ, заданный операндом A, выключается;
- I — логический ключ инвертируется, т. е. состояние его

меняется на противоположное, например, если был включен, будет выключен.

- 1) проверка состояния логического ключа с помощью блока gate

Блок GATE имеет такой же формат, как и при проверке состояний ОКУ и МКУ, и такие же два режима работы:

gate X A, [B]

Операнд A — имя или номер проверяемого ключа. Может быть именем, положительным целым числом, выражением в скобках, СЧА или СЧА*параметр.

Операнд B — метка блока, к которому будет направлен транзакт в случае невыполнения условия, заданного условным оператором X.

Условный оператор может принимать значения:

- LS — равен 1, если логический ключ, заданный операндом A, включен; 0 — если выключен;
- LR равен 1, если логический ключ, заданный операндом A, выключен; 0 — если включен.

```
sist      storage 7
generate  2,,,5
gate sv klu,met1
gate snf   sist,met1
enter     sist,3
advance   5
leave     sist,3
terminate 1
met1      terminate 1
generate  ,,1
logic s klu
```

```

met2      advance      7
          logic r klu
          gate se sist
          advance      1
          logic s klu
          transfer      ,met2
          2) проверка состояния логического ключа с помощью блока test и
          булевой переменной:

```

При проверке состояния логического ключа блок TEST имеет такой же формат записи и те же условные операторы, что и при проверке состояний ОКУ и МКУ. В булевой переменной используется только СЧА LS. Если в булевой переменной записать LR, возникнет ошибка «Предполагается фактор выражения».

Решение типовых задач

МКУ переводится в недоступное состояние через случайные интервалы времени. Для организации недоступности МКУ используем логический ключ и блок GATE, а не блоки SUNAVAIL и SAVAIL. Так как емкость МКУ Sist равна 8 и не делится на 3 без остатка, то блок GATE будет пропускать транзакт при недостаточной для удовлетворения запроса свободной емкости МКУ. Восстановление доступности МКУ начинается после того, как МКУ будет пусто.

```

zap      equ      3
sist      storage8
ver      fvariable  n$per/n$vse
          generate  (exponential(310,0,10.3))
vse      gate ls klu,met1
          gate snf  sist,met1
          enter     sist, zap
          advance   (normal(231,25.3,1.7))
          leave     sist, zap
per      terminate
          generate   ,,1
          logic s klu
met2      advance   (exponential(11,0,600.7))
          logic r klu
          gate se sist
          advance   3.2,1.3
          logic s klu
          transfer   ,met2
met1      terminate
          generate   3600
          savevalue  ver, v$ver
          terminate  1

```

Задачи для самостоятельного решения

1. В вышеприведённом примере блок gate в 5 строке будет пропускать транзакты при условии, что у МКУ недостаточно средств для обслуживания данных транзактов, и они становятся в список задержки МКУ (если операнд $B \neq 1$ и ёмкость МКУ не делится нацело на B; в случаях, если ёмкость делится на B нацело или $B=1$, список задержки МКУ всегда пуст). Модифицировать программу таким образом, чтобы транзакты не вставали в список задержки МКУ, а обрабатывались дублирующим МКУ ёмкостью 9;

2. Ответить на вопросы преподавателя по теоретической части.

Вопросы для обсуждения

1. Опишите блок logic.
2. Назовите состояния, в котором может находиться логический ключ.
3. Сделайте вывод по данной лабораторной работе

Оборудование и материалы: для выполнения данного практического занятия необходим компьютер с установленной операционной системой Windows 7 и программным продуктом: GPSS

Указания по технике безопасности: к выполнению практических работ допускаются студенты, ознакомившиеся с правилами работы в лаборатории, прошедшие инструктаж безопасности.

Задания: для выполнения практической работы необходимо выполнить следующее:

1. Изучить рекомендуемую литературу.
2. Выполнить практическую работу.
3. Ответить на контрольные вопросы.
4. Оформить отчет.

Содержание отчета: отчет по практической работе должен быть выполнен в редакторе MS Word и оформлен согласно требованиям. Требования по форматированию: Шрифт TimesNewRoman, интервал – полуторный, поля левое – 3 см., правое – 1,5 см., верхнее и нижнее – 2 см. Абзацный отступ – 1,25. Текст должен быть выровнен по ширине.

Отчет должен содержать титульный лист с темой практической работы, цель работы и описанный процесс выполнения вашей работы. В конце отчета приводятся выводы о проделанной работе.

В отчет необходимо вставлять скриншоты выполненной работы и добавлять описание к ним. Каждый рисунок должен располагаться по центру страницы, иметь подпись (Рисунок 1 – Создание подсистемы) и ссылку на него в тексте.

Список использованной литературы

1. Девянин П.Н. Управление доступом и информационными потоками: учебное пособие для студентов высших учебных заведений, обучающихся по специальностям направления подготовки 090300 - "Информационная безопасность вычислительных, автоматизированных и телекоммуникационных систем" и направлению подготовки 090900 - "Информационная безопасность" / П. Н. Девянин. Москва, 2011.

2. Гулятьев А. Визуальное моделирование в среде MATLAB. –СПб: Питер, 2010. – 432 с.

3. Емельянов В.В., Майорова В.И., Разумцова Ю.В. и др. Принятие оптимальных решений в интеллектуальных имитационных системах: Учебное пособие по курсам «Методы системного анализа и синтеза» и «Моделирование технологических и производственных процессов» – М.: Изд-во МГТУ им. Н.Э. Баумана, – 2008.

4. Кобелев Н.Б. Основы имитационного моделирования сложных экономических систем: Учеб. Пособие. – М.: Дело, 2009.

5. Советов Б.Я. Моделирование систем. Практикум: Учебное пособие для вузов/ Б.Я. Советов, С.А. Яковлев. – 2-е изд., перераб. и доп. – М.: Высш. шк., 2009.

6. Советов Б.Я., Яковлев С.А. Моделирование систем: учебник для вузов. – М.: Высш. шк., 2010.

Дополнительная:

7. Вендров А.М. Проектирование программного обеспечения экономических информационных систем. Учеб. - М.: Финансы и статистика. – 2010.

8. Девянин П.Н. Модели безопасности компьютерных систем. Учеб. пособие для

студентов вузов, обучающихся по специальности 075200 "Компьютер. безопасность" и 075500 "Комплекс. обеспечение информ. безопасности автоматизир. систем" - Москва, Сер. Высшее профессиональное образование. Информационная безопасность, 2005.

9. Емельянов А.А. и др. Имитационное моделирование экономических процессов: Учебное пособие / А.А. Емельянов, Е.А. Власова, Р.В. Дума; Под ред. А.А. Емельянова. – М.: Финансы и статистика, 2009.

10. Карпов Ю. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. - СПб.: БХВ-Петербург, 2005.

11. Кельтон В.Д., Лоу А.М. Имитационное моделирование. Классика CS. 3-е изд. - СПб.: Питер; Киев: Издательская группа BHV, 2004.

12. Семенов М.Г. Введение в математическое моделирование. М.: СОЛОН-Р, 2002.

Практическая работа № 8

Изучение функций.

Цель работы: изучить теоретическую часть, ознакомиться с нижеописываемыми блоками; выполнить практическую часть.

Формируемые компетенции: ПК-1, ПК-2, ПК-3, ПК-7, ПК-37

Теоретическая часть

1. Типы функций

Для применения в GPSS функция должна быть задана таблицей. Правила поиска значения функции по таблице определяет команда `function`, за которой следуют пары возможных значений аргумента и функции. Команда `function` имеет следующий формат:

Name function A, B

Name — имя функции.

Операнд A — аргумент функции. Может быть именем, положительным целым числом, строкой, выражением в скобках, СЧА или СЧА*параметр.

Операнд B — состоит из одной буквы, определяющей тип функции, и целого положительного числа, задающего количество пар возможных значений аргумента и функции (количество точек функции).

В GPSS World можно применять следующие пять типов функций:

- непрерывная числовая;
- D - дискретная числовая;
- E - дискретная атрибутивная;
- L - списковая числовая;
- M - списковая атрибутивная.

Существует еще один тип функции, которая применяется в блоках `generate` и `advance` в качестве операнда B. Она называется модификатор – функция. Значение этой функции вычисляется с двойной точностью и умножается на значение операнда A. Полученный результат используется как требуемая временная задержка в данном блоке.

Значения, заданные командой `function`, хранятся в форме чисел с плавающей запятой двойной точности. Эти значения ограничены 306 десятичными разрядами с точностью 15 десятичных разрядов.

2. Методы розыгрыша псевдослучайных чисел для дискретных равномерных распределений

Ранее было рассмотрено, как получить интервалы поступления и обслуживания заявок, распределенные по равномерному закону. Напомним, что в блоках `GENERATE` и `ADVANCE` равномерный закон распределения задается с помощью модификатора-интервала (см. п/з №1). Например:

`generate 3600, 900`

`advance 3600, 900`

Можно использовать также встроенный генератор дискретно – равномерного распределения `duniform(n, min, max)` (Приложение №4), где n – номер генератора равномерно распределённых случайных чисел (п/з №1, п. 6), min и max – минимальное и максимальное генерируемые значения соответственно.

`generate (duniform(1,2700,4500))`

`advance (duniform(1,2700,4500))`

Теперь используем оба этих метода на примерах и сравним полученные результаты.

`generate        3600, 900`

`queue           1`

`seize   Can`

```

depart 1
advance      3600, 900
release Can
terminate
generate      144000
terminate      1
-----
generate      (duniform(1, 2700, 4500))
queue         1
seize  Can
depart 1
advance      (duniform(1, 2700, 4500))
release Can
terminate
generate      144000
terminate      1

```

При использовании модификатора – интервала за 10000 прогонов блок generate сгенерировал 400005 транзактов, из которых ОКУ Can было обслужено 399981. Применение встроенного генератора `duniform(1, 2700, 4500)` дало 399926 и 399863 соответственно. Относительная разница составляет 0,0002. В обоих примерах в качестве генератора равномерно распределённых случайных чисел использовался RN1.

В моделях требуется получать и другие случайные величины, а не только интервалы поступления и обслуживания заявок, например, количество транзактов в сообщении, вычислительную сложность алгоритма и т. д. Для розыгрыша таких значений также можно использовать встроенный генератор дискретно – равномерного распределения.

```

Znak          fvariable      duniform(211, 5000, 9000)
.....
          assign      2, v$Znak
или
          assign      2, (duniform(211, 5000, 9000))

```

Для получения равномерно распределённых случайных чисел можно применять СЧА RN. Как уже отмечалось, СЧА RN принимает значения от 0 до 999 включительно. Значение, соответствующее равномерному распределению $A \pm B$, рекомендуется вычислять как выражение

$A - B + (2 \cdot B \cdot \text{RNn}) / 999$

Данное выражение получается путём добавления к меньшему значению выборки «случайной доли» ширины выборки. Наименьшим значением является $A - B$, а ширина выборки равна $2B$. Значение, полученное от генератора случайных чисел, рассматривается как «случайная доля». Таким образом получается новое значение $A - B + \text{RNn} \cdot 2 \cdot B$. Однако возникает следующая проблема. Новое значение может совпасть с максимально допустимым только в том случае, если RNn выдаст значение 1000. Но так как это невозможно, может быть только 999, то максимальное разыгрываемое значение $A + 2 \cdot B$ никогда не получить. Для исправления этого и рекомендуется использовать выражение

$A - B + (2 \cdot B \cdot \text{RNn}) / 999$. Используем его в следующем примере:

```

sred          equ          3600
stotk          equ          900
dravr          fvariable    sred-stotk+(2#stotk#rn733)/999
ran            fvariable    0
ran            equ          1
generate      ,,10000000

```

```

savevalue 1,v$dravr
test e (sred-stotk),x1,met1
terminate 1
met1 test e (sred+stotk),x1,met2
terminate 1
met2 terminate 1
start 10000000

```

Блок generate генерирует 10000000 транзактов. В блоке savevalue вычисляется арифметическая переменная dravr и полученное значение записывается в сохраняемую ячейку ran, которой командой equ присвоен номер 1. Первым блоком test проверяется равенство значения арифметической переменной минимальному генерируемому значению дискретного равномерного распределения, а вторым блоком test – максимальному генерируемому значению.

	START TIME	END TIME	BLOCKS	FACILITIES
STORAGES				
	0.000	0.000	7	0
0				

NAME	VALUE
DRAVR	10002.000
MET1	5.000
MET2	7.000
RAN	1.000
SRED	3600.000
STOTK	900.000

LABEL	LOC	BLOCK TYPE	ENTRY COUNT	CURRENT
COUNT RETRY				
	1	GENERATE	10000000	
0 0				
	2	SAVEVALUE	10000000	
0 0				
	3	TEST	10000000	
0 0				
	4	TERMINATE	10004	
0 0				
MET1	5	TEST	9989996	
0 0				
	6	TERMINATE	9991	
0 0				
MET2	7	TERMINATE	9980005	
0 0				

SAVEVALUE	RETRY	VALUE
1	0	2950.450

В первый и второй блоки terminate вошли соответственно 10004 и 9991 транзактов соответственно, т. е. 0,0001 в каждый. Точно такой же результат будет получен, если вместо переменной dravr использовать просто RN733, что подтверждает целесообразность выражения A-B+(2#B#RNn)/999.

В рассмотренных случаях результатами розыгрыша являлись дробные числа. В некоторых случаях требуется получить целые числа, например, при розыгрыше числа контейнеров, количества заявок в сообщении, количества деталей, поступающих на обработку, автомобилей в транспорте и т. д. Среди библиотечных арифметических функций (Приложение №4) язык GPSS World имеет функцию $\text{int}(A)$ выделения целой части с отбрасыванием дробной части. Тогда в приведённом выше примере при использовании данной функции переменная dravr выглядела бы так:

$\text{dravr} \quad \text{fvariable} \quad \text{int}(\text{sred-stotk} + (2\#\text{stotk}\#\text{rn733})/999)$

а в случае использования библиотечного генератора duniform – так:

$\text{dravr} \quad \text{fvariable} \quad \text{int}(\text{duniform}(211, 5000, 9000))$

3. Методы розыгрыша псевдослучайных чисел для дискретно – неравномерных распределений

При определении интервалов поступления и обслуживания заявок или других величин часто необходимо бывает разыграть числа для неравномерных распределений.

Пусть функция, которую нужно применить в модели, задана таблицей №1.

Таблица №1. Задание функции

Значение случайной величины	Относительная частота	Суммарная частота	Диапазон	Интервал
2	0,15	0,15	0,0. . 0,15	1
5	0,2	0,35	0,15. . 0,35	2
8	0,25	0,6	0,35. . 0,6	3
9	0,22	0,82	0,6. . 0,82	4
12	0,18	1,0	0,82. . 1,0	5

Согласно этой таблице случайная величина должна получать значения 2, 5, 8, 9 и 12 с частотами 0,15; 0,2; 0,25; 0,22; 0,18 соответственно. Графическая интерпретация функции представлена на рис. 1.

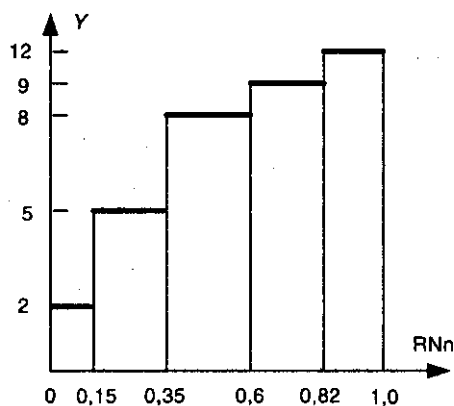


Рис.1 – Графическая интерпретация дискретной функции

Разыграем число в интервале $[0, 1]$. Пусть им будет число 0,523664. Оно попадает в интервал 3, соответствующий следующему значению случайной величины – 8. Таким образом, когда необходимо разыграть случайное число в соответствии с некоторым дискретным распределением, планировщик должен автоматически выполнить процедуру просмотра таблицы, которой задана функция. В данном случае для этого ему необходимо иметь:

- источник случайных чисел в интервале $[0, 1]$;
- значения случайной величины;
- суммарную частоту принятия значений случайной величиной.

Эти данные пользователь задаёт с помощью функции типа D – дискретного типа. В

списке данных функции типа D значения аргумента X должны быть типа Expression, а значения функции Y – типа integer, real или Name. Для таблицы №1 функция будет задана так:

```
Rasp      function      rn6, d5
.15,2/.35,5/.6,8/.82,9/1,12
```

Функция имеет имя Rasp, в качестве источника выступает rn6. Случайная величина может иметь 5 различных значений. Суммарные частоты и соответствующие им 5 значений функции записаны как пять пар чисел.

При написании пар чисел, определяющих координаты X и Y точки не только дискретной, но и функции любого типа, необходимо выполнять следующие правила:

- значения координат X и Y одной точки разделяется запятой;
- наборы координат разделяются знаком / или переводом строки;
- каждое последующее значение X должно быть больше предыдущего;
- каждая функция должна иметь, как минимум, две описанные таким образом точки;
- количество пар данных, указанных в операнде B команды function, должно совпадать с количеством пар, разделённых символом / в списке данных;
- списки данных функции не имеют полей комментария.

Как известно, блок transfer имеет такие режимы работы, как выборочный и функциональный. В режиме pick выбор блока, к которому направляется активный транзакт, производится случайно. Блоки должны быть расположены друг за другом и переходы к каждому из них равновероятны. Если они расположены не друг за другом, то следует использовать функцию и блок transfer в режиме fn:

```
Per      function      rn13,d4
0.2,met4/0.5,met2/0.76,met3/1.met4
      generate      (exponential(305,0,17.3))
.....
      transfer      fn, per
.....
met1      .....
.....
met2      .....
.....
met3      .....
.....
met4      .....
.....
```

Блок generate генерирует транзакты. При входе транзакты в блок transfer вычисляется функция Per, аргументом которой является rn13. Выбирается значение функции. В данном примере значения функции – метки блоков. В соответствии с выбранной меткой и производится безусловный переход активного транзакта к блоку, отмеченному выбранной меткой.

Функцию типа D можно использовать в блоке generate для генерирования транзактов с дискретными произвольными интервалами:

```
dint      function      rn831, d4
0.15, 3.2/0.55, 4.7/0.75, 7.1/1, 1.5
      generate      fn$dint
      terminate      1
```

Блок generate генерирует транзакты с интервалами 3,2, 4,7, 2,1 и 1,5, вероятности появления которых составляют 0,15, 0,4, 0,2 и 0,25 соответственно. В этом можно убедиться

на приведённом примере, задавая в команде start операнд A равным 1, 2, 3 и 4 и анализируя время завершения моделирования.

Также функцию типа D можно применять в блоке advance. Часто необходимо блоком advance организовать на одном и том же ОКУ или МКУ обслуживание заявок с различными временными характеристиками. В этом случае в качестве аргумента функции можно использовать какой-либо параметр транзакта:

```
obs      function      p1,d4
1, 3.75/2, 4.3/3, 2.1/4, 5.6
      generate      fn$dint
      terminate      1
.....
      advance      fn$obs
.....
```

При выходе активного транзакта из блока advance вычисляется функция с именем obs, аргументом которой является параметр p1. Полученный результат используется и как время задержки.

Замечание:

1) Если в функции типа D аргумент np, его запись больше 1 или не в строгом порядке возрастания вызывает ошибку «Вероятность не может быть больше единицы» или «Ошибка: убывающие интегральные вероятности» соответственно;

2) Если в функции типа D аргумент не np, его запись не в строгом порядке возрастания не вызывает ошибку, т. е. не обнаруживается, и результаты функции получаются неверными. Например, в функции

```
obs      function      p3,d4
2.15, 3.2/3.55, 4.7/11.75, 7.1/4.2, 5.6
```

система ошибку пользователя не обнаруживает. При p3=4 будет выдано значение функции 7.1, что неверно.

4. Процедура розыгрыша псевдослучайных чисел при непрерывном равномерном распределении

Для получения случайных чисел при непрерывном равномерном распределении используется функция типа C. Непрерывная функция определяется символом C, записываемым в качестве первого символа операнда B. В списке данных функции X и Y должны быть типа integer или real.

```
vgobs      function      rn23,c5
0.1, 14.2/0.35, 30.4/0.6, 50/0.85, 64/1, 80
```

Вначале непрерывная функция вычисляется так же, как и дискретная. При обращении к функции разыгрывается случайное число, которое далее используется как аргумент функции. Затем просматривается таблица для определения интервала значений суммарной вероятности, на который выпало случайное число. Если функция была определена как дискретная, берут второй элемент соответствующей пары, его значение становится значением функции. Если функция определена как непрерывная, выполняется линейная интерполяция для пары точек X_i и X_{i+1} , находящихся по краям того интервала значений вероятности, в который попало случайное число X, являющееся аргументом. Интерполяция производится согласно выражению

$$Y(X) = Y(X_i) + \frac{Y(X_{i+1}) - Y(X_i)}{X_{i+1} - X_i} \cdot (X - X_i), i = \overline{1, m} \quad (1)$$

где m – число точек функции.

Пусть в приведённом ранее примере получили rn23=0,43. Это число попадает в интервал [0,36; 0,6]. Если бы функция была дискретной, её значение равнялось бы 50.

Непрерывная функция вычисляется с учетом интерполяции по формуле (1):

$$Y(0.43) = Y(0.36) + \frac{Y(0.6) - Y(0.36)}{0.6 - 0.36} (0.43 - 0.36) = 30.4 + \frac{50 - 30.4}{0.24} 0.07 = 36.1166$$

skor function v\$put,c3
 2.1, 18.3/4.6, 24.1/7.4, 32.6

В качестве аргумента функции используется арифметическая переменная с именем put. Когда функция вызывается, сначала вычисляется эта переменная. Если значение аргумента выходит за пределы области определения функции, т. е. меньше 2,1 и больше 7,4, то функции присваиваются значения в ближайших предельных точках: 18,3 и 32,6 соответственно. При попадании аргумента в область определения функции выполняется линейная интерполяция по формуле (1).

Ввиду того, что непрерывная функция задаётся с помощью линейной интерполяции, для всех точек на одном интервале значений суммарной частоты существует одинаковая вероятность их использования. Это делает процедуру розыгрыша случайной величины при равномерном непрерывном распределении чрезвычайно простой. Предположим, что некоторая случайная переменная распределена равномерно и непрерывно в интервале от 2 до 6, т. е. (2; 6). Вероятность того, что значение переменной меньше 2, равно 0, а вероятность того, что она меньше 6, равна 1. Эти два значения суммарной вероятности используются для определения непрерывной функции:

ww2 function m32,c2
 0, 2/1, 6

Несмотря на то, что значением второй пары является 6, функция никогда не получает значение 6, потому что СЧА m при вычислении функции никогда не может быть больше 0,999999. Таким образом, наибольшей целой частью получаемого значения является 5.

Преимущество непрерывной функции по сравнению с дискретной очевидно, если в результате розыгрыша требуется получать целые числа. Предположим, что нужно разыграть целые числа в интервале от 100 до 475. Функцию можно описать так:

ww5 function m61,c2
 0, 100/1, 476

Если в этих же целях использовать дискретную функцию, нужно будет рассчитать 376 пар значений суммарных частот и задать 376 пар для определения функции. В последнем приведённом примере во второй паре указано 1, 476 для получения максимального генерируемого значения.

5. Методы розыгрыша псевдослучайных чисел для непрерывных неравномерных распределений

В некоторых случаях возникает необходимость розыгрыша псевдослучайных чисел согласно некоторому непрерывному распределению, задаваемому изначально какой-либо кривой. Для генерации таких чисел используется метод кусочной аппроксимации плотности распределения.

Пусть случайная величина, которую требуется моделировать, задана плотностью распределения $f(Y)$ в интервале $[a, b]$. Аппроксимируем функцию $f(Y)$ (рис. 2):

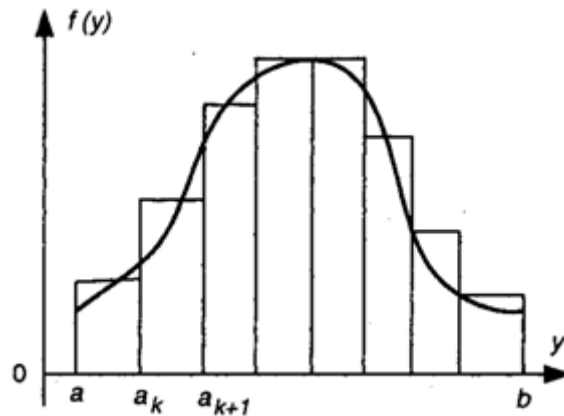


Рис. 2 – Кусочная аппроксимация плотности распределения

Разбивать $[a, b]$ на отрезки следует так, чтобы вероятность попадания случайной величины Y на любой из отрезков $[a_k, a_{k+1}]$ была постоянной, т. е. не зависела бы от расположения $[a, b]$. Если число отрезков m , то для любого отрезка $[a_k, a_{k+1}]$ должно быть выполнено равенство

$$\int_{a_i}^{a_{i+1}} f(Y) dY = \frac{1}{m},$$

на основе которого можно определить аппроксимирующее значение функции на отрезке $[a_k, a_{k+1}]$.

Процедура моделирования одного значения случайной величины Y содержит следующие этапы:

- формирование случайного числа с равномерным распределением в интервале $[0, 1]$;
- выбор по значению соответствующего ему отрезка $[a_k, a_{k+1}]$ на $[a, b]$, так, если $[a, b]$ содержит 10 отрезков, то значениям $0 \leq X_i \leq a_1$ соответствует первый отрезок, значениям $a_1 \leq X_i \leq a_2$ – второй отрезок и т. д.;
- вычисление значения случайной величины с заданным законом распределения $Y_i = a_k + (a_{k+1} - a_k) \cdot X_{i+1}$.

Рассмотрим применение метода кусочной аппроксимации к моделированию экспоненциального и нормального распределений.

1) моделирование экспоненциального распределения:

Когда имеет место пуассоновский поток (простейший поток, удовлетворяющий требованиям ординарности, стационарности, отсутствия последствия), поступление заявок описывается законом Пуассона, т. е. определяется уравнением

$$P_k(T) = \frac{(\lambda T)^k}{k!} e^{-\lambda T}, k = 0, 1, \dots,$$

где $P_k(T)$ – вероятность того, что k заявок придёт за время T , λ – средняя интенсивность поступления заявок.

При моделировании пуассоновского потока нет необходимости в определении вероятностей $P_k(T)$. Достаточно иметь лишь значения интервалов времени поступления требований. Расчёт значения времени прихода следующего транзакта в блок generate производится посредством сложения системного времени и разыгранной по определению интервалов времени величины. Это уравнение можно выразить относительно времени. Результатом является экспоненциальное уравнение.

Экспоненциальное распределение методом обратной функции может быть задано в такой форме, что если имеются равномерно распределённые интервале $[0, 1]$ числа, то

соответствующие интервалы времени могут быть вычислены непосредственно из выражения $T_p = T_{cp}[-\ln(1 - RNn)]$, где T_p – разыгранный интервал времени, T_{cp} – средний интервал времени, RNn – имя генератора равномерно распределённых случайных чисел. При использовании этого уравнения для моделирования пуассоновского входящего потока в блоке generate следует получить число от RNn и взять логарифм величины $(1 - RNn)$ со знаком минус и полученный результат умножить на T_{cp} .

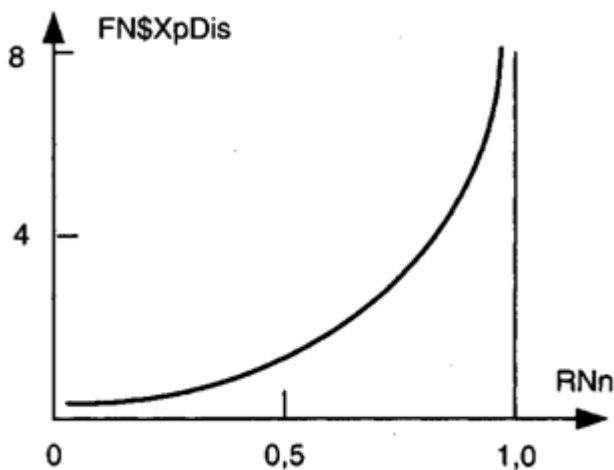
Первый этап можно выполнить путём задания функции, использующей в качестве аргумента RNn и принимающей значения $[-\ln(1 - RNn)]$. Функцию логарифма можно определить следующим способом. Распределение вида $[-\ln(1 - RNn)]$ следует аппроксимировать отрезками, накладываемыми на действительную функцию. Такая аппроксимация для GPSS осуществлена 23 отрезками, которые охватывают значения экспоненциальной функции случайной переменной на интервале $[0, 8]$. Определение такой функции, имеющей имя $XpDis$, состоит из 24 пар значений, используемых для преобразования значений RNn в $[-\ln(1 - RNn)]$:

$XpDis$ function rn200,c24
0,0/0.1,0.104/0.2,0.222/0.3,0.355/0.4,0.509/0.5,0.69/0.6,0.915/0.7,1.2/0.75/1.38/0.8,1.6
0.84,1.83/0.88,2.12/0.9,2.3/0.92,2.52/0.94,2.81/0.95,2.99/0.96,3.2/0.97,3.5/0.98,3.9
0.99,4.6/0.995,5.3/0.998,6.2/0.999,7/0.9998,8

Теоретически значения выражения $[-\ln(1 - RNn)]$ могут находиться в диапазоне от 0 до ∞ . Практически, поскольку наибольшим значением для RNn является 0,999999, наибольшим значением функции является 13,8155. Так как последним значением функции является пара 0.9998, 8, функция выдаёт в качестве своего значения величину 8, а не 13,8155 (рис. 3).

Рис. 3 – Графическая интерпретация функции $FN\$XpDis$

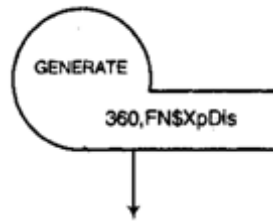
Однако RNn получает значения, большие, чем 0,9998, только в 0,0199% случаев. В большинстве случаев такая аппроксимация является вполне достаточной.



На втором этапе значение функции нужно умножить на T_{cp} , чтобы получить значение интервала времени, соответствующего пуассоновскому потоку. Для автоматического выполнения этого блок generate имеет специальное свойство. Пуассоновский поток моделируется с помощью блока generate путём выполнения следующих действий:

- в качестве операнда А используется значение интервалов времени;
- в качестве операнда В используется запись $FN\$XpDis$.

Значение интервала времени вычисляется в блоке посредством умножения значения функции на значение операнда А.



Предположим, что необходимо промоделировать пуассоновский входящий поток со средним значением интенсивности, равным 4, в течение каждых 24 часов. Единица времени – 1 минута. Когда транзакт входит в блок и переходит к следующему блоку, планировщик определяет время поступления следующего транзакта путём вычисления FN\$XpDis, умножения результат на 360 и добавления его к текущему времени таймера. Для вычисления функции FN\$XpDis необходимо определить значение RN23=0.35, FN\$XpDis=0.432 (среднее между 0,355 и 0,509). Произведение, равно 155,52, является искомым интервалом времени, прибавляемым к текущему значению таймера.

При использовании переменной, например
 rys fvariable 360#FN\$XpDis
 блоки

generate 360, FN\$XpDis

и

generate v\$rys

эквивалентны. Всё, что было изложено выше относительно моделирования пуассоновского потока блоком generate, остаётся в силе для моделирования экспоненциального времени задержки в блоке advance.

2) моделирование нормального распределения:

Случайная величина с нормальным распределением полностью описывается заданием двух значений: математического ожидания и стандартного отклонения. Нормированная случайная величина с нормальным распределением имеет математическое ожидание, равное 0, и стандартное отклонение, равное 1. Как правило, нормальные случайные величины, представляющие интерес, являются ненормированными. Для того, чтобы сделать выборку из ненормированного нормального распределения, удобнее сначала осуществить выборку из нормированного нормального распределения и произвести арифметические действия с полученной величиной, преобразовав её в ненормированную:

$$GNorm_{\text{выб}} = (GNorm_{\text{сот}}) \cdot (SNorm_{\text{выб}}) + GNorm_{\text{мож}},$$

где $SNorm_{\text{выб}}$ – величина, выбранная из нормированного нормального распределения, $GNorm_{\text{сот}}$ и $GNorm_{\text{мож}}$ – математическое ожидание и стандартное отклонение соответственно ненормированного нормального распределения.

Случайная величина для нормированного нормального распределения может быть найдена из выражения

$$RNn = \int_{-\infty}^{SNorm_{\text{выб}}} \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}} dX,$$

где RN_n – случайная величина любого из равномерных распределений GPSS от 0 до 1. Уравнение нельзя решить аналитически для нахождения точного значения $SNorm_{выб}$ для RN_n . Как следствие, величины $SNorm_{выб}$,

определяемые из уравнения для RN_n , должны быть протабулированы. На рис. 4 показана зависимость между RN_n и $SNorm_{выб}$.

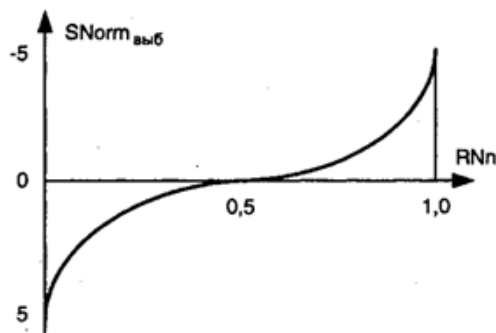


Рис. 4 – Графическая интерпретация $SNorm_{выб}$

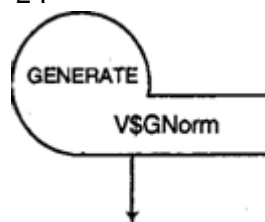
Эта кривая аппроксимируется с помощью непрерывной функции GPSS, составленной из 24 соответствующих ей прямолинейных сегментов. Определение функции, имеющей имя $snorm$, состоит из 25 пар значений:

```
snorm function      rn123,c25
0,-5/0.0003,-4/0.00135,-3/0.00621,-2.5/0.02275,-2/0.06681,-1.5/0.11507,-1.2
0.15866,-1/0.21186,-0.8/0.27245,-0.6/0.34458,-0.4/0.42074,-0.2/0.5,0.0/0.57926,0.2
0.65542,0.4/ 0.72575,0.6/0.78814,0.8/0.84134,1/0.88493,1.2/0.93319,1.5/0.97725,2
0.99379,2.5/0.99865,3/0.99997,4/1,5
```

Заметим, что, хотя величины $SNorm_{выб}$ могут лежать в пределах от $-\infty$ до $+\infty$, величины, взятые для $FN\$snorm$, могут меняться только в интервале от -5 до +5. Для практики это расхождение является приемлемым.

Предположим, что время между поступлениями заявок на передачу сообщений по каналу связи распределено по нормальному закону с математическим ожиданием 24 часа и стандартным отклонением 2 часа. За единицу времени принимается 1 час. Блок generate, имитирующий процесс поступления заявок, может быть представлен следующим образом. Переменная $GNorm$, используемая в операнде A блока generate, описана так:

```
GNorm      fvariable      2#fn$snorm+24
```



Таким образом, всякий раз, когда транзакт выходит из блока generate, планировщик случайным образом определяет величину времени выхода следующего транзакта, удовлетворяющего способу определения ненормированной случайной величины, распределённой по нормальному закону.

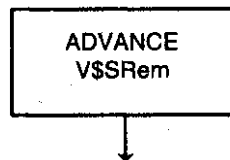
Пусть время обслуживания в некоторой точке модели имеет нормальное распределение, а математическое ожидание и стандартное отклонение в действительности зависят от вида осуществляемого обслуживания, например, ремонт различных типов средств связи в одном и том же ремонтном отделе одними и теми же мастерами (табл. 2).

Таблица №2 – Задание закона распределения в зависимости от типа обслуживания

Вид	Математическое	Стандартное
-----	----------------	-------------

обслуживания	ожидание	отклонение
1	30	5
2	20	3
3	36	6

Предположим, что вид обслуживания, необходимый транзакту, записан в его параметре 4. Тогда блок advance можно представить так:



Определения функций и арифметической переменной, обеспечивающих функционирование блока, записываются так:

```

mean      function      p4,d3
1,30/2,20/3,36
stde      function      p4,d3
1,5/2,3/3,6
SRem      fvariable      fn$stde#fn$snorm+fn$mean
  
```

Замечание:

Наименьшее значение функции равно -5. Может оказаться, что соответствующее значение выборки будет отрицательным. Однако на практике некоторые отрицательные величины не имеют смысла (например, время). Разработчик должен быть уверен, что математическое ожидание, по крайней мере, в 5 раз больше стандартного отклонения, иначе произойдёт остановка по ошибке «Отрицательное время задержки».

6. Функции типа E, L и M

1) Функция типа E:

Как известно, в дискретной числовой функции типа D значения аргумента X должны быть типа Expression, а значения функции Y – integer, real или Name. В дискретной атрибутивной функции типа E аргумент Y к тому же может быть СЧА и выражением в скобках.

```

vibor      function      p3,e4
1,v$norm1/7,v$norm2/12,r$usel/14,fn$sprav
  
```

Функция типа E вычисляется так же, как и функция типа D, с той лишь разницей, что после нахождения аргумента X вычисляется соответствующий СЧА или выражение в скобках.

Замечание:

Если в функции типа E Y имеет значения integer, real или Name, то она идентична функции типа D.

2) Функция типа L:

В функции типа E аргумент X может быть типа Expression, т. е. после вычисления принимать любые значения. В списковой числовой функции типа L аргумент X может быть типа integer, причём аргумент рассматривается как порядковый номер. Значения функции Y – integer, real или Name.

```

blok      function      q$dlna, l5
1,met1/2,met2/3,met3/4,met4/5,met5
  
```

По порядковому номеру находится значение функции. Если в результате вычисления аргумент меньше 1 или больше его заданного максимального значения, происходит остановка по ошибкам «Неположительный аргумент функции» и «Аргумент списковой функции слишком велик» соответственно. Значения аргумента должны начинаться с 1 и увеличиваться на 1 для каждой следующей пары данных. Если аргумент будет начинаться

не с 1 или будет нецелым, возникнет остановка по ошибке «Недопустимый порядковый номер в списке». В ходе моделирования вычисленное значение автоматически преобразовывается в целое, поэтому может иметь нецелые значения.

3) **Функция типа М:**

Списковая атрибутивная функция типа М отличается от списковой числовой функции типа L тем, что значения функции могут быть не только integer, real или Name, но и СЧА и выражением в скобках. Аргумент также целого типа и рассматривается как порядковый номер, поэтому он должен начинаться с 1 и возрастать на 1 для каждой следующей пары данных.

```
pos1      function      x$kvart,m5
1,q$stan/2,n$term3/3,s$usel/4,v$zirk/5,fn$susp
```

Вычисляется так же, как и функция типа L, только после нахождения аргумента – порядкового номера вычисляется соответствующий СЧА или выражение в скобках. Если в результате вычисления аргумент меньше 1 или больше его заданного максимального значения, происходит остановка по ошибкам «Неположительный аргумент функции» и «Аргумент списковой функции слишком велик» соответственно.

Замечание:

Если в функции типа М Y имеет значения типа integer, real или Name, то она аналогична функции типа L.

Решение типовых задач

На вход СМО с тремя каналами обслуживания и ограниченными длинами очередей поступает пуассоновский поток заявок со средним интервалом времени 1,5 минуты. Заявки могут быть трёх категорий. Вероятности поступления заявок 1, 2 и 3 категории 0.25, 0.45 и 0.3 соответственно. Среднее время обслуживания заявок составляет 3.5, 4.2 и 7 минут соответственно. Чем меньше среднее время обслуживания заявки, тем выше её приоритет. Построить имитационную модель с целью определения в течение 3 ч функционирования СМО среднего времени обслуживания заявками каждой категории, а также вероятностей их обслуживания и потерь в зависимости от длин очередей.

```
can3      storage 3
kat       function  rn5,d3
0.25,1/0.7,2/1,3
wrobs     function  p1,d3
1,210/2,252/3,420
dlina     function  p1,d3
1,4/2,4/3,4
class     function  p1,d3
1,1/2,2/3,3
vmod      equ      10800.0
srint     equ      90.0
;определение вероятностей обработки заявок
ver       fvariable n$wse/n$whod
ver1      fvariable n$term1/x1
ver2      fvariable n$term2/x2
ver3      fvariable n$term3/x3
;определение вероятностей потерь заявок
vpot f      variable n$spotwse/n$whod
vpot1     fvariable n$pot1/x1
vpot2     fvariable n$pot2/x2
vpot3     fvariable n$pot3/x3
generate  (exponential(3,0,srint))
```

```

whod      assign      1,fn$kat
          priorityfn$class
          savevalue    p1+,1 ;счёт заявок по категориям
          test e       p1,1,met3
          test l       q1,fn$dlina,potwse
          transfer     ,met5
met3       test e      p1,2,met4
          test l       q2,fn$dlina,potwse
          transfer     ,met5
met4       test l      q3,fn$dlina,potwse
met5       queue       oth ;встать в общую очередь
          queue        p1 ;встать в очереди по категориям
          enter        can3
          depart       oth ;выйти из общей очереди
          depart       p1 ;выйти из очереди по категориям
          advance      (exponential(1,0,fn$wrobs))
          leave        can3
wse        test e      p1,1,met1
term1      terminate
met1       test e      p1,2,term3
term2      terminate
term3      terminate
potwse     test e      p1,1,met2
pot1       terminate
met2       test e      p1,2,pot3
pot2       terminate
pot3       terminate
          generate     vmod
          savevalue    Ver11,v$ver1
          savevalue    Ver21,v$ver2
          savevalue    Ver31,v$ver3
          savevalue    Verwse,v$ver
          savevalue    pot11,v$vpot1
          savevalue    pot21,v$vpot2
          savevalue    pot31,v$vpot3
          savevalue    pot,v$vpot
          terminate    1
          start        1000,np ;np – не выводится отчёт
          clear
          start        5000

```

Результаты моделирования после 5000 прогонов:

QUEUE	MAX	CONT.	ENTRY	ENTRY(0)	AVE.CONT.	AVE.TIME	AVE.(-0)
RETRY							
1	4	3	95838	21032	2.127	1198.255	1535.149 0
2	4	3	219788	37799	1.538	377.958	456.460 0
3	4	2	177470	25011	0.478	145.351	169.195 0
OTH	12	8	493096	83842	4.143	453.673	546.615 0

STORAGE	CAP.	REM.	MIN.	MAX.	ENTRIES	AVL.	AVE.C.	UTIL.	RETRY
---------	------	------	------	------	---------	------	--------	-------	-------

DELAY

CAN3 3 0 0 3 493088 1 2.780 0.927 0 8

SAVEVALUE		RETRY	VALUE
1	0	150299.000	
2	0	270078.000	
3	0	179685.000	
VER11	0	0.638	
VER21	0	0.814	
VER31	0	0.988	
VERWSE	0	0.822	
POT11	0	0.362	
POT21	0	0.186	
POT31	0	0.012	
POT	0	0.178	

Задачи для самостоятельного решения

1. Пусть приходит пуассоновский поток со стандартным отклонением 15,3 мин заявок на ремонт высокоточных приборов 4 типов обслуживания: 1 – гарантийное обслуживание; 2 – незначительный ремонт; 3 – ремонт средней степени сложности; 4 – ремонт высокой степени сложности. Заявки каждого типа появляются с вероятностями 0.3, 0.35, 0.24 и 0.11 соответственно. Время ремонта прибора, прилагающегося к заявке каждого типа, каждого типа указано в таблице. Заявки случайным образом распределяются между 3 терминалами, тратящими на регистрацию 10 ± 7 минут. Далее приборы направляются в два ремонтных отдела, могущих одновременно обслуживать 4 прибора. 1 отдел занимается заявками 1 и 2 типа, 2 – 3 и 4 типа. Один из пяти приборов 4 типа признаётся не подлежащим ремонту и отправляется на запчасти (утилизируется). Чем легче требуемый вид обслуживания, тем выше приоритет заявки. Очередь к терминалам ограничена 6 посетителями на каждый, при превышении этого числа остальные не обслуживаются. Для каждой из очередей по типу заявки при превышении 20 единиц в каждой из них приходящие заявки отбрасываются. Требуется провести моделирование в течение 6 часов, определив число утилизированных приборов, вероятность появления таких приборов, среднее время ожидания заявок в очереди к терминалам и в очередях на ремонт. Также для каждого (при больших числах, как в приведённом примере – для 30) транзакта в матрицу записать время появления в модели, время вывода из модели, тип, номер терминала, номер отдела, величину времени, затраченную на ремонт прибора, метку утилизации;

	Тип заявки			
	1	2	3	4
Время ремонта, мин	10...25	35...55	60...100	110...190

Время ремонта распределено по непрерывному равномерному распределению;

Вопросы для обсуждения

1. Ответьте на вопросы преподавателя по теоретической части.
2. Какой формат имеет команда определения функции в GPSS World?

3. Какие типы функций существуют? Расскажите про каждый тип функций;
4. Какой библиотечный генератор и эквивалентное ему выражение используются для розыгрыша дискретных равномерных распределений?

Оборудование и материалы: для выполнения данного практического занятия необходим компьютер с установленной операционной системой Windows 7 и программным продуктом: GPSS

Указания по технике безопасности: к выполнению практических работ допускаются студенты, ознакомившиеся с правилами работы в лаборатории, прошедшие инструктаж безопасности.

Задания: для выполнения практической работы необходимо выполнить следующее:

1. Изучить рекомендуемую литературу.
2. Выполнить практическую работу.
3. Ответить на контрольные вопросы.
4. Оформить отчет.

Содержание отчета: отчет по практической работе должен быть выполнен в редакторе MS Word и оформлен согласно требованиям. Требования по форматированию: Шрифт TimesNewRoman, интервал – полуторный, поля левое – 3 см., правое – 1,5 см., верхнее и нижнее – 2 см. Абзацный отступ – 1,25. Текст должен быть выровнен по ширине.

Отчет должен содержать титульный лист с темой практической работы, цель работы и описанный процесс выполнения вашей работы. В конце отчета приводятся выводы о проделанной работе.

В отчет необходимо вставлять скриншоты выполненной работы и добавлять описание к ним. Каждый рисунок должен располагаться по центру страницы, иметь подпись (Рисунок 1 – Создание подсистемы) и ссылку на него в тексте.

Список использованной литературы

1. Девянин П.Н. Управление доступом и информационными потоками: учебное пособие для студентов высших учебных заведений, обучающихся по специальностям направления подготовки 090300 - "Информационная безопасность вычислительных, автоматизированных и телекоммуникационных систем" и направлению подготовки 090900 - "Информационная безопасность" / П. Н. Девянин. Москва, 2011.

2. Гуляев А. Визуальное моделирование в среде MATLAB. –СПб: Питер, 2010. – 432 с.

3. Емельянов В.В., Майорова В.И., Разумцова Ю.В. и др. Принятие оптимальных решений в интеллектуальных имитационных системах: Учебное пособие по курсам «Методы системного анализа и синтеза» и «Моделирование технологических и производственных процессов» – М.: Изд-во МГТУ им. Н.Э. Баумана, – 2008.

4. Кобелев Н.Б. Основы имитационного моделирования сложных экономических систем: Учеб. Пособие. – М.: Дело, 2009.

5. Советов Б.Я. Моделирование систем. Практикум: Учебное пособие для вузов/ Б.Я. Советов, С.А. Яковлев. – 2-е изд., перераб. и доп. – М.: Высш. шк., 2009.

6. Советов Б.Я., Яковлев С.А. Моделирование систем: учебник для вузов. – М.: Высш. шк., 2010.

Дополнительная:

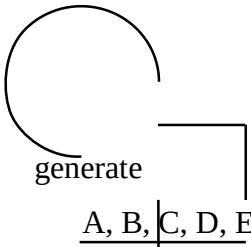
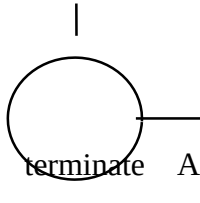
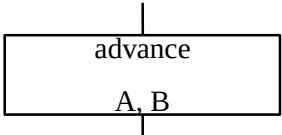
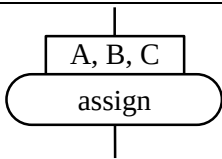
7. Вендров А.М. Проектирование программного обеспечения экономических информационных систем. Учеб. - М.: Финансы и статистика. – 2010.

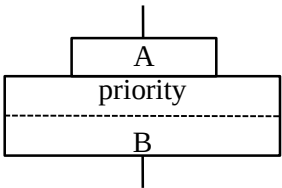
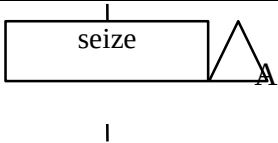
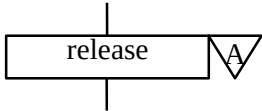
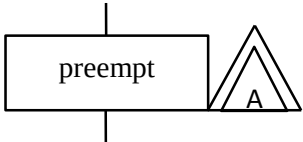
8. Девянин П.Н. Модели безопасности компьютерных систем. Учеб. пособие для студентов вузов, обучающихся по специальности 075200 "Компьютер. безопасность" и 075500 "Комплекс. обеспечение информ. безопасности автоматизир. систем" - Москва, Сер. Высшее профессиональное образование. Информационная безопасность, 2005.

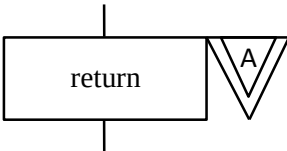
9. Емельянов А.А. и др. Имитационное моделирование экономических процессов: Учебное пособие / А.А. Емельянов, Е.А. Власова, Р.В. Дума; Под ред. А.А. Емельянова. – М.: Финансы и статистика, 2009.
10. Карпов Ю. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. - СПб.: БХВ-Петербург, 2005.
11. Кельтон В.Д., Лоу А.М. Имитационное моделирование. Классика CS. 3-е изд. - СПб.: Питер; Киев: Издательская группа BHV, 2004.
12. Семененко М.Г. Введение в математическое моделирование. М.: СОЛОН-Р, 2002.

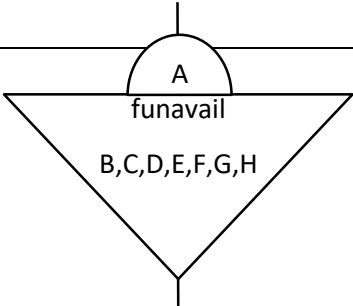
Приложение №1

Форматы и графическое изображение блоков

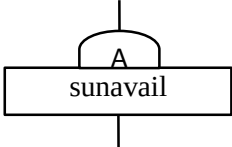
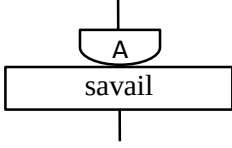
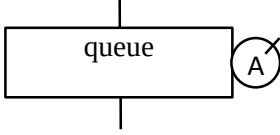
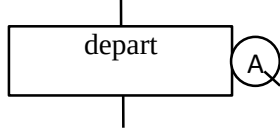
№	Название и формат	Операнды			Графическое изображение
		имя	значение	тип*	
1.	Источник потока транзактов. Формат: generate A, B, C, D, E	[A]	Среднее время	1...5, 10	
		[B]	Полуинтервал или модификатор - функция	1...5, 10	
		[C]	Время начальной задержки	1...5, 10	
		[D]	Количество генерируемых транзактов	1...5, 8, 10	
		[E]	Приоритет	1, 2, 4, 5, 8, 9	
2.	Удаление из модели активного транзакта. Формат: terminate A	[A]	Величина уменьшения счётчика завершения	1, 2, 5...8	
3.	Задержка транзакта на заданный отрезок модельного времени. Формат: advance A, B	A	Среднее время задержки	2...7	
		[B]	Полуинтервал или модификатор - функция	1...7	
4.	Присвоение или изменение значения параметра	A	Параметр активного транзакта	2...7	
		B	Значение	1...7	
		[C]	Номер функции	1, 2,	

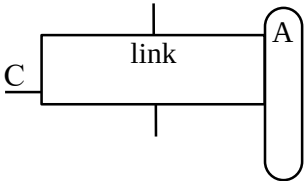
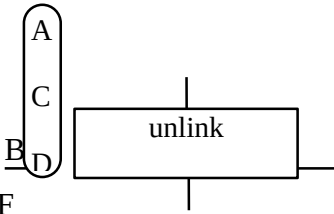
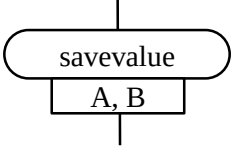
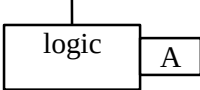
	транзакта. Формат: assign A[±], B, C			5...8	
5.	Установка приоритета активному транзакту. Формат: priority A, B	A	Новое значение приоритета	2, 4...7, 9	
		[B]	Помещение транзакта в СТС	1, BU	
6.	Занятие устройства. Формат: seize A	A	Имя или номер устройства	2, 5...8	
7.	Освобождение занятого устройства. Формат: release A	A	Имя или номер устройства	2, 5...8	
8.	Захват устройства. Формат: preempt A, B, C, D, E	A	Имя или номер устройства	2, 5...8	
		[B]	Режим	1, PR	
		[C]	Мерка блока	1, 2, 5...8	
		[D]	Имя или номер параметра	1, 2, 5...8	
		[E]	Режим удаления	RE	

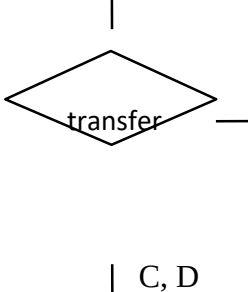
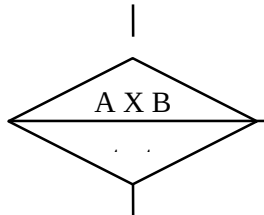
9.	Освобождение устройства от захвата. Формат: return A	A	Имя или номер устройства	2, 5...8	
----	---	---	--------------------------	----------	---

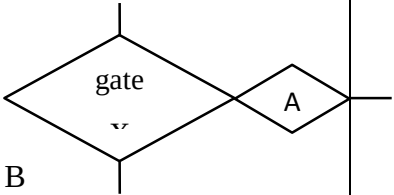
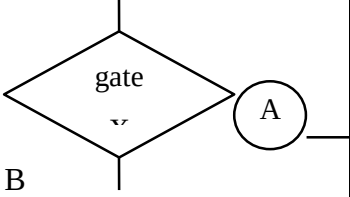
10.	Перевод	A	Имя или номер	2, 5...8	
-----	---------	---	---------------	----------	---

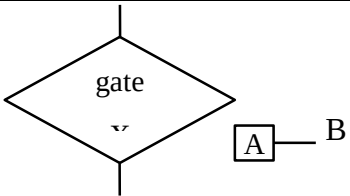
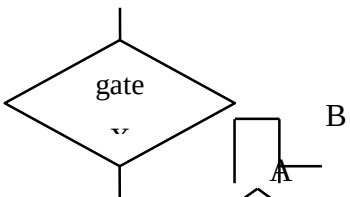
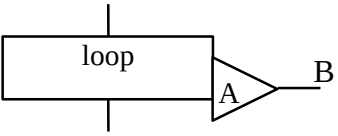
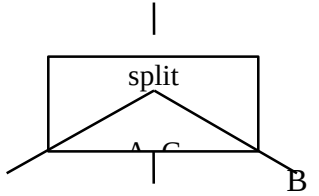
	устройства в недоступное состояние. Формат: funavail A, B, C, D, E, F, G, H		устройства		
		[B]	Режим удаления или продолжения	1, RE, CO	
		[C]	Метка блока выхода	1, 2, 5...8	
		[D]	Имя или параметр устройства	1, 2, 5...8	
		[E]	Режим удаления или продолжения	1, RE, CO	
		[F]	Метка блока выхода	1, 2, 5...8	
		[G]	Режим удаления или продолжения	1, RE, CO	
		[H]	Метка блока выхода	1, 2, 5...8	
11.	Перевод устройства в доступное состояние. Формат: favail A	A	Имя или номер устройства	2, 5...8	funavail
12.	Занятие памяти. Формат: enter A, B	A	Имя или номер памяти	2, 5...8	enter A
		[B]	Число занимаемых элементов памяти	1, 2, 5...8	
13.	Освобождение памяти.	A	Имя или номер памяти	2, 5...8	leave A

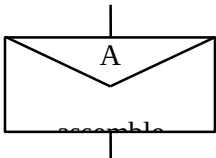
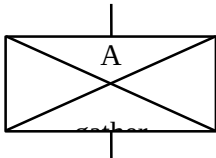
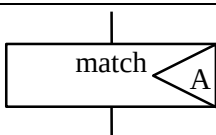
	Формат: leave A, B	[B]	Число освобождаемых элементов памяти	1, 2, 5...8	
14.	Перевод памяти в недоступное состояние. Формат: sunavail A	A	Имя или номер памяти	2, 5...8	
15.	Перевод памяти в доступное состояние. Формат: savail A	A	Имя или номер памяти	2, 5...8	
16.	Увеличение длины очереди. Формат: queue A, B	A	Имя или номер очереди	2, 5...8	
		[B]	Число, на которое увеличивается длина очереди	1, 2, 5...8	
17.	Уменьшение длины очереди. Формат: depart A, B	A	Имя или номер очереди	2, 5...8	
		[B]	Число, на которое уменьшается длина очереди	1, 2, 5...8	

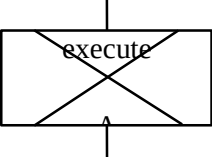
18.	Присоединение транзакта к списку пользователя. Формат: link A, B, C	A	Имя или номер списка пользователя	2, 5...8	
		B	Задание дисциплины обслуживания	5...7, LIFO, FIFO	
		[C]	Метка альтернативного блока	1, 2, 5...8	
19.	Удаление транзакта из списка пользователя. Формат: unlink X A, B, C, D, E, F	[X]	Условный оператор	1, G, GE, L, LE, E, NE	
		A	Имя или номер списка пользователя	2, 5...8	
		B	Метка блока выхода	2, 5...8	
		[C]	Число удаляемых транзактов	1, 2, 5...8, ALL	
		[D]	Имя или номер параметра	1, 2, 5...8, BACK	
		[E]	Аргумент для сравнения	1...7	
		[F]	Метка альтернативного блока	1, 2, 5...8	
20.	Модификация значения ячейки. Формат: savevalue A[±], B	A	Имя или номер ячейки	2, 5...8	
		B	Модификатор	2...7	
21.	Изменение состояния	X	Логический оператор	S, R, I	

	логического ключа. Формат: logic X A	A	Имя или номер логического ключа	2, 5...8	
22	Переход активного транзакта в указанный блок. Формат: transfer A, B, C, D	[A]	Режим перехода: безусловный; статистический; параметрически й; функциональны й; оба; все; выборочный; одновременный; подпрограммы й.	1, 2, 5...8 [,] .xxx P FN BOTH ALL PICK SIM SBR	
		[B]	Имя или номер блока (параметра, функции)	1, 2, 5...8	
		[C]	Имя или номер блока (аргумент для режимов P и FN)	1, 2, 5...8	
		[D]	Шаг нумерации блоков для режима ALL	1, 2, 5...8	
23	Управление потоком транзактов в зависимости от результата	X	Условный оператор	G, GE, L, LE, N, NE	
		A	Первая сравниваемая величина	2...7	

	сравнения операндов. Формат: test X A, B, C	B	Вторая сравниваемая величина	2...7	
		[C]	Метка блока	1, 2, 5...8	
24	Управление поток транзактов в зависимости от состояния устройства. Формат: gate X A, B	X	Проверяемое условие: занято или нет; захвачено или нет; доступно или нет.	U, NU, I, NI, FV, FNV	
		A	Имя или номер устройства	2, 5...8	
		[B]	Метка блока	1, 2, 5...8	
25	Управление поток транзактов в зависимости от состояния памяти. Формат: gate X A, B	X	Проверяемое условие: пусто или нет; заполнено или нет; доступно или нет.	SE, SNE, SF, SNF, SV, SNV	
		A	Имя или номер памяти	2, 5...8	
		[B]	Метка блока	1, 2, 5...8	

26.	Управление потоком транзактов в зависимости от состояния ключа. Формат: gate X A, B	X	Проверяемое условие: ключ установлен или нет.	LS, LR	
		A	Имя или номер логического ключа	2, 5...8	
		[B]	Метка блока	1, 2, 5...8	
27.	Управление потоком транзактов в зависимости от состояния синхронизации проверяемого транзакта. Формат: gate X A, B	X	Проверяемое условие: есть ли транзакт, ожидающий условия синхронизации в указанном блоке MATCH или нет.	M, NM	
		A	Метка блока MATCH	2, 5...8	
		[B]	Метка блока	1, 2, 5...8	
28.	Организация циклического прохождения транзактов в модели. Формат: loop A, B	A	Параметр, содержащий число циклов	2, 5...8	
		[B]	Метка блока – начало цикла	1, 2, 5...8	
29.	Создание необходимого числа копий (семейства) активного транзакта.	A	Количество создаваемых копий	2, 5...8	
		[B]	Метка блока для входа копий	1, 2, 5...8	
		[C]	Имя или номер	1, 2, 5...8	

	Формат: split A, B, C		параметра, в который помещаются порядковые номера копий		
30.	Объединение транзактов одного семейства. Формат: assemble A	A	Количество собираемых транзактов	2, 5...8	
31.	Накопление заданного числа транзактов одного семейства. Формат: gather A	A	Количество транзактов, которое необходимо накопить	2, 5...8	
32.	Синхронизация движения двух транзактов одного семейства. Формат: match A	A	Метка сопряженного блока MATCH	2, 5...8	

33.	Разрешение транзакту на выполнение операции любого другого блока. Формат: execute A	A	Метка блока, операция которого выполняется активным транзактом	2, 5...8	
-----	---	---	--	----------	---

Примечание

*В таблице допустимые значения операндов указаны цифрами, которые соответствуют:

- 1 - Null (нет ввода);
- 2 - Name (имя);
- 3 - Number (неотрицательное число);
- 4 - String (строковый);
- 5 - ParenthesizedExpression (выражение в скобках);
- 6 - SNA (СЧА);
- 7 - SNA*Parameter (косвенная адресация);
- 8 - PosInteger (положительное целое число);
- 9 - Integer (целое число со знаком (необязательно));
- 10 - DirectSNA (прямой СЧА, т. е. не использующий косвенную адресацию).

Приложение №2

Системные числовые атрибуты GPSS

№	Код	Значение	Примечание
Системные числовые атрибуты			
1	RNj	Случайное число. Целочисленное значение.	Датчик генерирует последовательность равномерно распределённых случайных целых чисел от 0 до 999.
2	C1	Относительное модельное время. Вещественное значение.	Модельное время с момента последней команды RESET. Автоматически изменяется системой.
3	AC1	Абсолютное модельное время. Вещественное значение.	Модельное время с момента последней команды CLEAR. Автоматически изменяется системой.
4	TG1	Число, равное текущему значению счётчика завершения. Целочисленное значение.	Начальное значение TG1 задаётся оператором START и указывает на завершение моделирования, когда становится равным нулю. Вошедшие в блоки TERMINATE с ненулевым операндом A уменьшают это значение на число, равное значению операнда A.
5	Z1	Текущая свободная системная память. Целочисленное значение.	
Системные числовые атрибуты транзактов			
6	Pj	Значение j-го параметра активного транзакта. Целочисленное, вещественное или строковое значение.	Для косвенной адресации используются: СЧА*j; СЧА*имя параметра; СЧА*\$ имя параметра.
7	XN1	Номер активного транзакта. Целочисленное значение.	
8	PR	Приоритет активного транзакта.	Может изменяться блоком

		Целочисленное значение.	PRIORITY. По умолчанию приоритет равен 0.
9	MBj	Соответствие в блоке MATCH. Целочисленное значение.	1, если в сопряжённом блоке MATCH находится транзакт того же семейства, что и активный транзакт; 0 – иначе.
10	MPj	Время прохождения транзактом некоторого участка модели. Вещественное значение.	Вычисляется как разность текущего абсолютного модельного времени и значения j-го параметра активного транзакта. Разность – транзитное время – сохраняется в j-м параметре.
11	M1	Время пребывания в модели активного транзакта. Вещественное значение.	Вычисляется как разность текущего абсолютного модельного времени и отметки времени активного транзакта, в которой хранится либо время поступления транзакта в модель, либо время последнего прохождения транзактом блока MARK без операнда A.
Системные числовые атрибуты блоков			
12	Nj	Общее число входов транзактов в j-й блок. Целочисленное значение.	Увеличивается при каждом входе транзакта в j-й блок.
13	Wj	Текущее число транзактов, находящихся в j-м блоке. Целочисленное значение.	Увеличивается при каждом входе транзакта в j-й блок и уменьшается при каждом выходе транзакта из j-го блока.
Системные числовые атрибуты МКУ (памяти)			
14	Sj	Текущее содержимое памяти j. Целочисленное значение.	Изменяется блоками ENTER и LEAVE.
15	Rj	Число свободных единиц памяти j. Целочисленное значение.	Изменяется блоками ENTER и LEAVE.
16	SRj	Коэффициент использования памяти j. Вещественное значение.	Выражается в тысячных долях, но значение СЧА – целое, т. е.

			коэффициент равен 0,65, то SRj равен 650. Может быть нецелочисленным.
17	SAj	Среднее значение занятой памяти j. Вещественное значение.	Взвешенное по времени среднее количество занятых элементов памяти.
18	SMj	Максимальное количество элементов занятой памяти j. Целочисленное значение.	Максимальное количество элементов памяти, которое было занято одновременно в процессе моделирования.

11 -

19	SCj	Счётчик использования памяти j. Целочисленное значение.	Общее количество использовавшихся элементов памяти j. Увеличивается при каждом входе транзакта в блок ENTER.
20	STj	Среднее время использования одного элемента памяти j.	Определяется системой автоматически.
21	SEj	Занятость памяти j.	1 – не занята; 0 – занята.
22	SFj	Заполненность памяти j.	1 – заполнена; 0 – не заполнена.
23	SVj	Доступность памяти j.	1 – доступна; 0 – недоступна.
Системные числовые атрибуты ОКУ			
24	Fj	Занятость устройства j. Целочисленное значение.	1 – занято; 0 – не занято. Может быть изменён блоками SEIZE, RELEASE, PREEMPT и RETURN.
25	Flj	Прерванность устройства j. Целочисленное значение.	1 – прервано; 0 – не прервано. Может быть изменён блоками PREEMPT и RETURN.
26	FVj	Доступность устройства j. Целочисленное значение.	1 – доступно; 0 – недоступно. Может быть изменён блоками FUNAVAIL и FAVAIL.
27	FRj	Коэффициент использования устройства j.	Выражается в тысячных долях, но значение СЧА – целое, т. е. коэффициент равен 0,88, то FRj равен 880.

28	FCj	Количество транзактов, занимавших устройство j.	Увеличивается при входе транзакта в блок SEIZE или PREEMPT.
29	FTj	Среднее время использования устройства j одним транзактом.	Определяется системой автоматически.
Системные числовые атрибуты очередей			
30	Qj	Текущая длина очереди j. Целочисленное значение.	Увеличивается при входе транзакта в блок QUEUE и уменьшается при входе в блок DEPART.
31	QAj	Средняя длина очереди j. Вещественное значение.	Взвешенное во времени количество транзактов в очереди j.
32	QMj	Максимальная длина очереди j. Целочисленное значение.	Максимальное количество транзактов, находившихся одновременно в очереди j.
33	QCj	Общее число входов в очередь j. Целочисленное значение.	Автоматически увеличивается при входе транзакта в очередь j.
34	QZj	Число входов с нулевым временем пребывания в очереди j.	Число входов транзактов, время нахождения которых в очереди равно нулю.
35	QTj	Среднее время пребывания транзакта в очереди j.	Рассчитывается с учётом нулевых входов.
36	QXj	Среднее время пребывания транзакта в очереди j.	Рассчитывается без учёта нулевых входов.
Системные числовые атрибуты ячеек и матриц ячеек			
37	Xj	Содержимое ячейки j. Целочисленное, вещественное или строковое значение.	Изменяется блоком SAVEVALUE.
38	MXj (a, b)	Содержимое ячейки матрицы j. Целочисленное, вещественное или строковое значение.	Значение ячейки в строке a и столбце b. Изменяется блоком SAVEVALUE. Для матриц, имеющих более двух измерений, все остальные индексы считаются равными 1. В отличие от СЧА класса MX, PLUS- выражения могут обращаться к любому

			элементу многомерных матриц.
Системные числовые атрибуты вычислительных объектов			
39	FNj	Результат вычисления значения функции j. Вещественное значение.	
40	Vj	Результат вычисления целочисленной переменной j или переменной j с плавающей точкой. Вещественное значение.	
41	BVj	Результат вычисления булевой переменной j. Вещественное значение.	Принимает значения: 1(true); 0 (false).
Системные числовые атрибуты групп, списков и ключей			
42	GNj	Текущее число членов в j-й числовой группе.	
43	GTj	Текущее число членов в j-й группе транзактов.	
44	CHj	Текущее число транзактов в j-м списке пользователя.	Список пользователя создаётся и ведётся самим программистом в отличие от остальных списков модели. Используются блоки LINK и UNLINK. Значения всех СЧА определяются аналогично значениям СЧА очередей.

12 -

45	CAj	Среднее число транзактов в j-м списке пользователя.	
46	CMj	Максимальное число транзактов в j-м списке пользователя.	
47	CCj	Общее число транзактов в j-м списке пользователя.	
48	CTj	Среднее время пребывания транзакта в j-м списке пользователя.	

49	LSj	Состояние j-го логического ключа.	Принимает значения: 1- установлен; 0 – не установлен.
Системные числовые атрибуты таблиц			
50	TBj	Среднее значение невзвешенных аргументов таблицы j.	Для занесения в таблицу используется блок TABULATE.
51	TCj	Число включений в таблицу j.	Целочисленное значение.
52	TDj	Среднеквадратическое отклонение для таблицы j.	Вещественное значение.

Приложение №3

Допустимые значения операндов операторов блоков GPSS

№ п/ п	Оператор блока	Операнд	Null	Name	Number	String	Parenthesiz edExpression	SNA	SNA*Para- meter	PosInteger	Integer	DirectSNA	Другие значения	По умолчанию
			1	2	3	4	5	6	7	8	9	10	11	12
1	adopt	A		+			+	+	+	+				
2	advance	A		+	+	+	+	+	+					
		B	+	+	+	+	+	+	+					
3	alter	0	Null, условные операторы: E, G, GE,L, LE, MAX, MIN, NE											
		A		+			+	+	+	+				
		B	+	+			+	+	+	+				ALL
		C		+			+	+	+	+			PR	
		D		+	+	+	+	+	+					
		E	+	+			+	+	+	+			PR	
		F	+	+	+	+	+	+	+					
		G	+	+			+	+	+	+				
4	assemble	A		+			+	+	+	+				
5	assign	A		+			+	+	+	+				
		B		+	+	+	+	+	+					
		C	+	+			+	+	+	+				
6	close	A	+	+			+	+	+	+				
		B	+	+			+	+	+	+				1
		C	+	+			+	+	+	+				
7	count	0	Логические операторы: FNV, FV, I, LS, NI, NU, SE, SF, SNE, SNF, SNV, SV, U; условные операторы: E, G, GE,L, LE, MAX, MIN, NE											
		A		+			+	+	+	+				
		B		+			+	+	+	+				
		C		+			+	+	+	+				
		D	+	+	+	+	+	+	+					

		E	+										+	
8	depart	A		+			+	+	+	+				
		B	+	+		+	+	+	+	+				1
9	displace	A		+			+	+	+	+				
		B		+			+	+	+	+				
		C	+	+			+	+	+	+				
		D	+	+			+	+	+	+				
10	enter	A		+			+	+	+	+				
		B	+	+			+	+	+	+				
11	examine	A		+			+	+	+	+				
		B	+	+	+	+	+	+	+					
		C		+			+	+	+	+				
12	execute	A		+			+	+	+	+				
13	favail	A		+			+	+	+	+				
14	funavail	A		+			+	+	+	+				
		B	+										RE, CO	
		C	+	+			+	+	+	+				
		D	+	+			+	+	+	+				
		E	+										RE, CO	
		F	+	+			+	+	+	+				
		G	+										RE, CO	
		H	+	+			+	+	+	+				
15	gate	A		+			+	+	+	+				
		B	+	+			+	+	+	+				
16	gather	A		+			+	+	+	+				
17	generate	A	+	+	+	+	+					+		
		B	+	+	+	+	+					+		
		C	+	+	+	+	+					+		
		D	+	+	+	+	+			+		+		

		E	+	+		+	+			+	+			
18	index	A		+			+	+	+	+				
		B		+	+		+	+	+					
19	integrati on	A		+										
		B	+										ON, OFF	
20	join	A		+			+	+	+	+				
		B	+	+	+	+	+	+	+	+				
21	leave	A		+			+	+	+	+				
		B	+		+			+	+	+				1
22	link	A		+			+	+	+	+				
		B					+	+	+				LIFO, FIFO	
		C	+	+			+	+	+	+				
23	logic	0	Логические операторы: S, R, I											
		A	+											
24	loop	A		+			+	+	+	+				
		B	+	+			+	+	+	+				
25	mark	A	+	+			+	+	+	+				
26	match	A		+			+	+	+	+				
27	msaveva lue	A		+			+	+	+	+				
		B		+			+	+	+	+				
		C		+			+	+	+	+				
		D		+			+	+	+	+				
28	open	A		+		+	+	+	+	+				
		B	+	+			+	+	+	+				
		C	+	+			+	+	+	+				
29	plus	A		+	+	+	+	+	+					
		B	+	+			+	+	+	+				
30	preempt	A		+			+	+	+	+				

		B	+										PR	
		C	+	+			+	+	+	+				
		D	+	+			+	+	+	+				
		E											RE	
31	priority	A		+			+	+	+		+			
		B	+										BU	
32	queue	A		+			+	+	+	+				
		B	+	+			+	+	+	+				
33	read	A		+			+	+	+	+				
		B	+	+			+	+	+	+				1
		C	+	+			+	+	+	+				
34	release	A		+			+	+	+	+				
35	remove	0	Null, условные операторы: E, G, GE,L, LE, MAX, MIN, NE											E
		A		+			+	+	+	+				
		B	+	+			+	+	+	+			ALL	
		C	+	+	+	+	+	+	+					
		D	+	+			+	+	+	+			PR	
		E	+	+	+	+	+	+	+					
		F	+	+			+	+	+	+				
36	return	A		+			+	+	+	+				
37	savail	A		+			+	+	+	+				
38	savevalu e	A		+			+	+	+	+				
		B	+	+	+	+	+	+	+	+				
39	scan	0	Null, условные операторы: E, G, GE,L, LE, MAX, MIN, NE											E
		A		+			+	+	+	+				
		B	+	+			+	+	+	+			PR	
		C	+	+	+	+	+	+	+					
		D		+			+	+	+	+			PR	
		E		+			+	+	+	+				
		F	+	+			+	+	+	+				
40	seek	A		+			+	+	+	+				

		B	+	+			+	+	+	+				
41	seize	A		+			+	+	+	+				
42	split	A		+			+	+	+	+				
		B	+	+			+	+	+	+				
		C	+	+			+	+	+	+				
43	select	0	Логические операторы: FNV, FV, I, LS, NI, NU, SE, SF, SNE, SNF, SNV, SV, U; условные операторы: E, G, GE,L, LE, MAX, MIN, NE											
		A		+			+	+	+	+				
		B		+			+	+	+	+				
		C	+	+			+	+	+	+				
		D		+	+	+	+	+	+					
		E	+											
		F	+	+			+	+	+	+				
44	sunavail	A		+			+	+	+	+				
45	tabulate	A		+			+	+	+	+				
		B	+	+	+		+	+	+					
46	terminat e	A	+	+			+	+	+	+				
47	test	0	Условные операторы: E, G, GE,L, LE, MAX, MIN, NE											
		A		+	+	+	+	+	+					
		B		+	+	+	+	+	+					
		C	+	+			+	+	+	+				
48	transfer	A	+	+			+	+	+	+			+	
		B	+	+			+	+	+	+				
		C	+	+			+	+	+	+				
		D	+	+			+	+	+	+				
49	unlink	0	Null, условные операторы: E, G, GE,L, LE, MAX, MIN, NE											E
		A		+			+	+	+	+				
		B		+			+	+	+	+				
		C	+	+			+	+	+	+			ALL	
		D	+	+			+	+	+	+			BAC	

[illegible]

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Методические указания
по организации и проведению самостоятельной работы
по дисциплине **«ПРОГРАММНО-АППАРАТНЫЕ СРЕДСТВА
ЗАЩИТЫ ИНФОРМАЦИИ»**
для студентов специальности 10.05.01 Компьютерная безопасность
специализация «Информационно-аналитическая и техническая экспертиза
компьютерных систем»

Ставрополь
2026

ВВЕДЕНИЕ

Целью изучения дисциплины "Программно-аппаратные средства защиты информации" является формирование набора общепрофессиональных, профессиональных и профессионально-специализированной компетенций студента по специальности 10.05.01 Компьютерная безопасность.

Общая характеристика самостоятельной работы студента

Основными видами учебной работы по достижению результатов освоения дисциплины являются лекции, практические задания и самостоятельная работа студентов (СРС).

На лекциях раскрываются основные положения и понятия курса, формируются знания в области документоведения.

На лабораторных работах формируются умения, необходимые для решения задач профессиональной деятельности.

Приступая к изучению учебной дисциплины, необходимо ознакомиться с учебной программой, получить в библиотеке рекомендованные учебные пособия, а также получить у преподавателя в электронном виде конспекты лекций, методические рекомендации к практическим заданиям, завести новую тетрадь для конспектирования лекций и выполнения практических и работ.

Для изучения дисциплины предлагается список основной и дополнительной литературы. Основная литература предназначена для обязательного изучения, дополнительная – поможет более глубоко освоить отдельные вопросы, подготовить исследовательские задания и выполнить задания для самостоятельной работы.

В ходе лекционных занятий студент обязан осуществлять конспектирование учебного материала, особое внимание, обращая на категории, формулировки, раскрывающие содержание тех или иных понятий, физических явлений, процессов, эффектов. В рабочих конспектах желательно оставлять поля, на которых следует делать пометки из рекомендованной литературы, дополнять материал прослушанной лекции, формулировать научные выводы и практические рекомендации. Студент имеет право задавать преподавателю уточняющие вопросы с целью уяснения теоретических положений, разрешения спорных ситуаций.

Самостоятельная работа студентов над материалом учебной дисциплины является неотъемлемой частью учебного процесса и должна предполагать углубление знания учебного материала, излагаемого на аудиторных заданиях, и приобретение дополнительных знаний по отдельным вопросам самостоятельно.

Основными видами самостоятельной работы студентов по учебной дисциплине являются:

- самостоятельное выполнение заданий по заданным темам;

- самостоятельное изучение учебного материала по заданным темам;
- подготовка к практическим заданиям, оформление отчета по выполненному практическому занятию и подготовка к собеседованию по результатам работы.

Основными методами самостоятельной работы студентов являются:

- 1) для овладения знаниями:
 - чтение текста (конспекта лекций, учебника, дополнительной литературы) и конспектирование текста;
 - работа с нормативными документами;
 - поиск информации в Интернет;
- 2) для закрепления и систематизации знаний:
 - работа с конспектом лекции (обработка текста);
 - повторная работа над учебным материалом (учебника, дополнительной литературы, слайдами);
 - составление плана и тезисов ответа или графическое изображение структуры ответа;
 - составление таблиц для систематизации учебного материала;
 - изучение нормативных документов;
 - ответы на контрольные вопросы;
- 3) для формирования умений:
 - подготовка к практическим заданиям;
 - решение задач и практических заданий;
 - подготовка отчетов по практическим заданиям;
 - рефлексивный анализ профессиональных умений.

В случае пропуска учебного задания студент может воспользоваться содержанием различных блоков учебно-методического комплекса (лекции, практические задания, контрольные вопросы и т.д.) для самоподготовки и освоения темы. Для самоконтроля необходимо использовать вопросы и задания, предлагаемые к практическим заданиям, а также варианты тестовых заданий.

Технологическая карта самостоятельной работы студента

Коды реализуемых компетенций, индикатора (ов)	Вид деятельности студентов	Средства и технологии оценки	Объем часов, в том числе		
			СРС	Контактная работа с преподавателем	Всего
10 семестр					
ОПК-15 ИД-1	Выполнение разноуровневых заданий, решение разноуровневых задач	Индивидуальное задание	4,75	0.25	5

ОПК-15 ИД-1	Перевод статьи из иностранного источника по заданной тематике	Текст перевода	4,75	0.25	5
ОПК-15 ИД-1	Подготовка к коллоквиуму	Коллоквиум	9.50	0.50	10.00
ОПК-15 ИД-1	Подготовка к практическому занятию	Собеседование	0.95	0.125	2.5
Итого за 10 семестр			21,375	1,125	22.5
Итого					

Порядок выполнения самостоятельной работы

Методические рекомендации по работе с учебной литературой

При работе с книгой необходимо подобрать литературу, научиться правильно ее читать, вести записи. Для подбора литературы в библиотеке используются алфавитный и систематический каталоги.

Важно помнить, что рациональные навыки работы с книгой – это всегда большая экономия времени и сил.

Правильный подбор учебников рекомендуется преподавателем, читающим лекционный курс. Необходимая литература может быть также указана в методических разработках по данному курсу.

Изучая материал по учебнику, следует переходить к следующему вопросу только после правильного уяснения предыдущего, описывая на бумаге все выкладки и вычисления (в том числе те, которые в учебнике опущены или на лекции даны для самостоятельного вывода).

При изучении любой дисциплины большую и важную роль играет самостоятельная индивидуальная работа.

Особое внимание следует обратить на определение основных понятий курса. Студент должен подробно разбирать примеры, которые поясняют такие определения, и уметь строить аналогичные примеры самостоятельно. Нужно добиваться точного представления о том, что изучаешь. Полезно составлять опорные конспекты. При изучении материала по учебнику полезно в тетради (на специально отведенных полях) дополнять конспект лекций. Там же следует отмечать вопросы, выделенные студентом для консультации с преподавателем.

Выводы, полученные в результате изучения, рекомендуется в конспекте выделять, чтобы они при перечитывании записей лучше запоминались.

Опыт показывает, что многим студентам помогает составление листа опорных сигналов, содержащего важнейшие и наиболее часто употребляемые формулы и понятия. Такой лист помогает запомнить формулы, основные положения лекции, а также может служить постоянным справочником для студента.

Чтение научного текста является частью познавательной деятельности. Ее цель – извлечение из текста необходимой информации. От того на сколько осознанна читающим собственная внутренняя установка при обращении к печатному слову (найти нужные сведения, усвоить информацию полностью

или частично, критически проанализировать материал и т.п.) во многом зависит эффективность осуществляемого действия.

Выделяют четыре основные установки в чтении научного текста:

информационно-поисковый (задача – найти, выделить искомую информацию)

усваивающая (усилия читателя направлены на то, чтобы как можно полнее осознать и запомнить, как сами сведения, излагаемые автором, так и всю логику его рассуждений)

аналитико-критическая (читатель стремится критически осмыслить материал, проанализировав его, определив свое отношение к нему)

творческая (создает у читателя готовность в том или ином виде – как отправной пункт для своих рассуждений, как образ для действия по аналогии и т.п. – использовать суждения автора, ход его мыслей, результат наблюдения, разработанную методику, дополнить их, подвергнуть новой проверке).

Основные виды систематизированной записи прочитанного:

Аннотирование – предельно краткое связное описание просмотренной или прочитанной книги (статьи), ее содержания, источников, характера и назначения;

Планирование – краткая логическая организация текста, раскрывающая содержание и структуру изучаемого материала;

Тезирование – лаконичное воспроизведение основных утверждений автора без привлечения фактического материала;

Цитирование – дословное выписывание из текста выдержек, извлечений, наиболее существенно отражающих ту или иную мысль автора;

Конспектирование – краткое и последовательное изложение содержания прочитанного.

Конспект – сложный способ изложения содержания книги или статьи в логической последовательности. Конспект аккумулирует в себе предыдущие виды записи, позволяет всесторонне охватить содержание книги, статьи. Поэтому умение составлять план, тезисы, делать выписки и другие записи определяет и технологию составления конспекта.

Методические рекомендации по составлению конспекта:

1. Внимательно прочитайте текст. Уточните в справочной литературе непонятные слова. При записи не забудьте вынести справочные данные на поля конспекта;

2. Выделите главное, составьте план;

3. Кратко сформулируйте основные положения текста, отметьте аргументацию автора;

4. Законспектируйте материал, четко следуя пунктам плана. При конспектировании старайтесь выразить мысль своими словами. Записи следует вести четко, ясно.

5. Грамотно записывайте цитаты. Цитируя, учитывайте лаконичность, значимость мысли.

В тексте конспекта желательно приводить не только тезисные положения, но и их доказательства. При оформлении конспекта необходимо стремиться к емкости каждого предложения. Мысли автора книги следует излагать кратко, заботясь о стиле и выразительности написанного. Число дополнительных элементов конспекта должно быть логически обоснованным, записи должны распределяться в определенной последовательности, отвечающей логической структуре произведения. Для уточнения и дополнения необходимо оставлять поля.

Овладение навыками конспектирования требует от студента целеустремленности, повседневной самостоятельной работы.

Вопросы для коллоквиумов

- Lock»
1. Система защиты информации от несанкционированного доступа «Dallas
 2. Запуск и регистрация в системе защиты
 3. Создание пользователей
 4. Реализация мандатной модели разграничения доступа
 5. Реализация дискреционной модели разграничения доступа
 6. Обеспечение замкнутости программной среды
 7. Контроль целостности
 8. Регистрация событий
 9. Печать штампа
 10. Гарантированное удаление данных
 11. Реализация запрета загрузки ПЭВМ в обход СЗИ
 12. Система защиты информации «Secret NET 5.0-C»
 13. Общие сведения
 14. Запуск и регистрация в системе защиты.
 15. Создание учетных записей пользователей
 16. Реализация дискреционной модели разграничения доступа
 17. Реализация мандатной модели разграничения доступа
 18. Режим замкнутой программной среды
 19. Контроль целостности
 20. Регистрация событий
 21. Печать штампа
 22. Гарантированное удаление данных
 23. Настройка механизма шифрования
 24. Средства организации виртуальных частных сетей
 25. Задачи, решаемые VPN
 26. Туннелирование в VPN
 27. Уровни защищенных каналов
 28. Защита данных на канальном уровне
 29. Организация VPN средствами протокола PPTP
 30. Постановка задачи
 31. Установка и настройка VPN
 32. Анализ защищенности передаваемой информации
 33. Защита данных на сетевом уровне
 34. Протокол SKIP
 35. Протокол IPSec
 36. Организация VPN средствами СЗИ VipNet
 37. Настройка сетевых соединений виртуальных машин
 38. Установка СЗИ VipNet

Повышенный уровень:

1. Протокол SKIP
2. Протокол IPSec
3. Организация VPN средствами СЗИ VipNet
4. Настройка сетевых соединений виртуальных машин
5. Установка СЗИ VipNet
6. Настройка СЗИ VipNet
7. Использование протокола IPSec для защиты сетей
8. Шифрование трафика с использованием протокола IPSec
9. Проверка защиты трафика
10. Настройка СЗИ VipNet

11. Использование протокола IPSec для защиты сетей
12. Шифрование трафика с использованием протокола IPSec
13. Проверка защиты трафика

1. Критерии оценивания компетенций

Оценка «отлично» выставляется студенту, если он на высоком уровне: анализирует компьютерную систему с целью определения уровня защищенности и доверия; Применяет инструментальные средства проведения сертификационных испытаний; составляет и оформляет аналитический отчет по проведенным сертификационным испытаниям; делать выводы по оценке защищенности на основании аналитического отчета.

Оценка «хорошо» выставляется студенту, если на среднем уровне: анализирует компьютерную систему с целью определения уровня защищенности и доверия; Применяет инструментальные средства проведения сертификационных испытаний; составляет и оформляет аналитический отчет по проведенным сертификационным испытаниям; делать выводы по оценке защищенности на основании аналитического отчета.

Оценка «удовлетворительно» выставляется студенту, если на минимальном уровне: анализирует компьютерную систему с целью определения уровня защищенности и доверия; Применяет инструментальные средства проведения сертификационных испытаний; составляет и оформляет аналитический отчет по проведенным сертификационным испытаниям; делать выводы по оценке защищенности на основании аналитического отчета.

Оценка «неудовлетворительно» выставляется студенту, если он с грубыми ошибками: анализирует компьютерную систему с целью определения уровня защищенности и доверия; Применяет инструментальные средства проведения сертификационных испытаний; составляет и оформляет аналитический отчет по проведенным сертификационным испытаниям; делать выводы по оценке защищенности на основании аналитического отчета.

Оценка «отлично» выставляется студенту, если он на высоком уровне: разрабатывает программы и методики аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; проведения аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; подготовки заключения по результатам аттестации объектов вычислительной техники на соответствие требованиям по защите информации.

Оценка «хорошо» выставляется студенту, если на среднем уровне: разрабатывает программы и методики аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; проведения аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; подготовки заключения по результатам аттестации объектов вычислительной техники на соответствие требованиям по защите информации.

Оценка «удовлетворительно» выставляется студенту, если на минимальном уровне: разрабатывает программы и методики аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; проведения аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; подготовки заключения по результатам аттестации объектов вычислительной техники на соответствие требованиям по защите информации.

Оценка «неудовлетворительно» выставляется студенту, если он с грубыми ошибками: разрабатывает программы и методики аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; проведения аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; подготовки заключения по результатам аттестации объектов вычислительной техники на соответствие требованиям по защите информации.

Оценка «отлично» выставляется студенту, если он на высоком уровне: проводит аттестационные испытания защищенности объектов вычислительной техники; проведения

экспериментальных исследований уровней защищенности объектов вычислительной техники.

Оценка «хорошо» выставляется студенту, если на среднем уровне: проводит аттестационные испытания защищенности объектов вычислительной техники; проведения экспериментальных исследований уровней защищенности объектов вычислительной техники.

Оценка «удовлетворительно» выставляется студенту, если на минимальном уровне: проводит аттестационные испытания защищенности объектов вычислительной техники; проведения экспериментальных исследований уровней защищенности объектов вычислительной техники.

Оценка «неудовлетворительно» выставляется студенту, если он с грубыми ошибками: проводит аттестационные испытания защищенности объектов вычислительной техники; проведения экспериментальных исследований уровней защищенности объектов вычислительной техники.

2. Описание шкалы оценивания

Максимально возможный балл за весь текущий контроль устанавливается равным 55. Текущее контрольное мероприятие считается сданным, если студент получил за него не менее 60% от установленного для этого контроля максимального балла. Рейтинговый балл, выставляемый студенту за текущее контрольное мероприятие, сданное студентом в установленные графиком контрольных мероприятий сроки, определяется следующим образом:

Уровень выполнения контрольного задания	Рейтинговый балл (в % от максимального балла за контрольное задание)
Отличный	100
Хороший	80
Удовлетворительный	60
Неудовлетворительный	0

3. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

Процедура проведения данного оценочного мероприятия включает в себя:

- устный ответ по вопросам из указанных тем;

Предлагаемые студенту вопросы позволяют проверить ИД-2 оПК-16, ИД-1 ПК-2, ИД-1 ПК-3, ИД-2 ПК-3 компетенции.

Для подготовки к данному оценочному мероприятию необходимо изучить рекомендованные источники и научную литературу и на их основании подготовиться к указанным в методических материалах вопросам для обсуждения. Для подготовки к данному оценочному мероприятию необходимо 10 часов, в течение которых обучающийся изучает научную и учебную литературу, готовится к сдаче коллоквиума.

При подготовке к ответу студенту предоставляется право пользования конспектом.

При проверке задания, оцениваются умение последовательно, четко и логически стройно излагать избранную проблему, обобщать, сопоставлять различные точки зрения по рассматриваемой проблеме, аргументировать основные положения и выводы, использовать научную литературу.

Оценочный лист студента (ки) _____ Ф.И.О.		
№ гр. _____		
Оценка складывается как среднее арифметическое из трех оценок: правильность ответа; умение приводить примеры; умение отвечать на дополнительные вопросы		
Оценка правильности ответа	Оценка умения приводить примеры	Оценка умения отвечать на дополнительные вопросы
Итоговая оценка		

Тематика для написания статьи, тезисов Базовый уровень

1. Методики оценки защищенности и противодействия кибератакам в системах индустриального Интернета вещей на основе онтологии метрик безопасности и методов интеллектуального анализа больших данных
2. Решение для предотвращения фишинга с использованием человеческого поведения и эмоций, влияющих на успех фишинга
3. Моделирование риска безопасности промышленной киберфизической системы при кибератаках
4. Оценка риска атаки и защиты с использованием концепции культуры кибербезопасности
5. Метод оценки рисков безопасности контейнера пограничных вычислений на основе теории игр
6. Защита информационной безопасности компьютерной сети энергосистемы
7. Оценка рисков кибербезопасности морских портов: пример контейнерного порта
8. Метод оценки ситуации сетевой безопасности, основанный на глубоком обучении.
9. Аналитика больших данных на графах кибератак для определения приоритетов и требований безопасности
10. Приложение для автоматизации оценивания безопасности в корпоративной сети
11. Оценка рисков безопасности и защищенности в киберфизических системах
12. Калькулятор рисков информационной безопасности мобильных устройств
13. Анализ безопасности промышленной сети управления и принятие решений
14. Анализ соответствующих процессов безопасности для снижения риска в избирательной системе
15. Построение системы индикаторов оценки обстановки информационной безопасности

Повышенный уровень:

16. Построение системы индикаторов оценки обстановки по широте и долготе информационной безопасности
17. Измерение межпроцессного взаимодействия в распределенных системах
18. Исследование кибератак на ключевое контрольно-измерительное оборудование в энергосистеме
19. Средство оценки времени до компрометации на основе модели для оценки состояния безопасности уязвимых сетей.
20. Устойчивость интеллектуальных транспортных систем

1. Критерии оценивания компетенций

Оценка «отлично» выставляется студенту, если он на высоком уровне: анализирует компьютерную систему с целью определения уровня защищенности и доверия; Применяет инструментальные средства проведения сертификационных испытаний; составляет и оформляет аналитический отчет по проведенным сертификационным испытаниям; делает выводы по оценке защищенности на основании аналитического отчета.

Оценка «хорошо» выставляется студенту, если на среднем уровне: анализирует компьютерную систему с целью определения уровня защищенности и доверия; Применяет инструментальные средства проведения сертификационных испытаний; составляет и

оформляет аналитический отчет по проведенным сертификационным испытаниям; делать выводы по оценке защищенности на основании аналитического отчета.

Оценка «удовлетворительно» выставляется студенту, если на минимальном уровне: анализирует компьютерную систему с целью определения уровня защищенности и доверия; Применяет инструментальные средства проведения сертификационных испытаний; составляет и оформляет аналитический отчет по проведенным сертификационным испытаниям; делать выводы по оценке защищенности на основании аналитического отчета.

Оценка «неудовлетворительно» выставляется студенту, если он с грубыми ошибками: анализирует компьютерную систему с целью определения уровня защищенности и доверия; Применяет инструментальные средства проведения сертификационных испытаний; составляет и оформляет аналитический отчет по проведенным сертификационным испытаниям; делать выводы по оценке защищенности на основании аналитического отчета.

Оценка «отлично» выставляется студенту, если он на высоком уровне: разрабатывает программы и методики аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; проведения аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; подготовки заключения по результатам аттестации объектов вычислительной техники на соответствие требованиям по защите информации.

Оценка «хорошо» выставляется студенту, если на среднем уровне: разрабатывает программы и методики аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; проведения аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; подготовки заключения по результатам аттестации объектов вычислительной техники на соответствие требованиям по защите информации.

Оценка «удовлетворительно» выставляется студенту, если на минимальном уровне: разрабатывает программы и методики аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; проведения аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; подготовки заключения по результатам аттестации объектов вычислительной техники на соответствие требованиям по защите информации.

Оценка «неудовлетворительно» выставляется студенту, если он с грубыми ошибками: разрабатывает программы и методики аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; проведения аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; подготовки заключения по результатам аттестации объектов вычислительной техники на соответствие требованиям по защите информации.

Оценка «отлично» выставляется студенту, если он на высоком уровне: проводит аттестационные испытания защищенности объектов вычислительной техники; проведения экспериментальных исследований уровней защищенности объектов вычислительной техники.

Оценка «хорошо» выставляется студенту, если на среднем уровне: проводит аттестационные испытания защищенности объектов вычислительной техники; проведения экспериментальных исследований уровней защищенности объектов вычислительной техники.

Оценка «удовлетворительно» выставляется студенту, если на минимальном уровне: проводит аттестационные испытания защищенности объектов вычислительной техники; проведения экспериментальных исследований уровней защищенности объектов вычислительной техники.

Оценка «неудовлетворительно» выставляется студенту, если он с грубыми ошибками: проводит аттестационные испытания защищенности объектов вычислительной

техники; проведения экспериментальных исследований уровней защищенности объектов вычислительной техники.

2. Описание шкалы оценивания

Максимально возможный балл за весь текущий контроль устанавливается равным 55. Текущее контрольное мероприятие считается сданным, если студент получил за него не менее 60% от установленного для этого контроля максимального балла. Рейтинговый балл, выставляемый студенту за текущее контрольное мероприятие, сданное студентом в установленные графиком контрольных мероприятий сроки, определяется следующим образом:

Уровень выполнения контрольного задания	Рейтинговый балл (в % от максимального балла за контрольное задание)
Отличный	100
Хороший	80
Удовлетворительный	60
Неудовлетворительный	0

3. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

Процедура проведения данного оценочного мероприятия включает в себя:

- проверка правильности написания тезисов по одной из указанных тем;

Предлагаемые студенту вопросы позволяют проверить ИД-2 ОПК-16, ИД-1 ПК-2, ИД-1 ПК-3, ИД-2 ПК-3 компетенции.

Для подготовки к данному оценочному мероприятию необходимо изучить рекомендованные источники и научную литературу и на их основании подготовить тезисы по указанным в методических материалах вопросам. Для подготовки к данному оценочному мероприятию необходимо 10 часов, в течение которых обучающийся изучает научную и учебную литературу, составляет тезисы.

При подготовке к ответу студенту предоставляется право пользования любыми источниками научной литературы.

При проверке задания, оцениваются умение последовательно, четко и логически стройно излагать избранную проблему, обобщать, сопоставлять различные точки зрения по рассматриваемой проблеме, аргументировать основные положения и выводы, использовать научную литературу.

Оценочный лист студента (ки) _____ Ф.И.О.		
№ гр. _____		
Оценка складывается как среднее арифметическое из трех оценок: правильность ответа; умение приводить примеры; умение отвечать на дополнительные вопросы		
Оценка правильности ответа	Оценка умения приводить примеры	Оценка умения отвечать на дополнительные вопросы
		Итоговая оценка

Темы докладов Базовый уровень

1. Мотивы действий компьютерных хакеров.
2. Использование средств активной разведки в компьютерных системах и сетях.
3. Средства атак на пароли в системах аутентификации.
4. Российские стандарты в области средств идентификации и аутентификации.
5. Международные стандарты в области средств идентификации и аутентификации.
6. Методы социальной инженерии в компьютерных системах.
7. Эволюция парольной аутентификации в ОС Unix.
8. Обзор стандартов и руководств по управлению паролями.
9. Способы управления выбором паролей.
10. Способы организации словарных атак на пароли.
11. Способы хранения паролей в компьютерных системах.
12. Средства атаки на пароли BIOS.
13. Методы атак на устройства аутентификации.
14. Основные причины успеха компьютерных преступлений.
15. Эволюция методов аутентификации в ОС Windows.
16. Средства аутентификации в компьютерных системах Apple Macintosh.
17. Эволюция алгоритмов симметричной криптографии и атак на них.
18. Методы шифрования на основе условного (депонированного) ключа.
19. Способы усиления парольной аутентификации.
20. Сертифицированные средства шифрования файлов.
21. Сертифицированные средства шифрования дисков.
22. Возможности пользователей по запоминанию сложных паролей.
23. Обзор и сравнительный анализ средств хранения парольных баз данных.
24. Эволюция методов и средств биометрической аутентификации.
25. Атаки на системы биометрической аутентификации.
26. Способы защиты от подбора паролей.
27. Обзор средств «запоминания» паролей в сетевых службах.
28. Особенности биометрической аутентификации.
29. Обзор средств шифрования информации на флэш-дисках.
30. Обзор функций хеширования паролей в ОС и СУБД.
31. Обзор средств генерации паролей.
32. Обзор средств выявления «слабых» паролей.
33. Обзор средств фильтрации паролей.
34. Использование биометрии в идентификационных картах и паспортах.
35. Практическое применение биометрии в России.
36. Другая тема, выбранная студентом и согласованная с преподавателем.

Повышенный уровень

1. Средства аутентификации по отпечаткам пальцев.
2. Средства аутентификации по форме руки.
3. Средства аутентификации по характеристикам глаза.
4. Средства аутентификации по лицу.
5. Средства аутентификации по клавиатурному почерку.
6. Средства аутентификации по голосу.
7. Средства аутентификации по рукописной подписи.
8. Социальные и правовые проблемы использования биометрической аутентификации.
9. Программы для подбора телефонных номеров.
10. Адресация в сетях X.25 и АТМ.
11. Обзор известных атак с вызовом отказа в обслуживании.

12. Сравнительный анализ протоколов создания виртуальных частных сетей.
13. Обзор атак на банкоматы и пластиковые карты.
14. Устройства аутентификации iButton (Touch Memory).
15. USB-устройства аутентификации.
16. Устройства аутентификации – генераторы одноразовых паролей.
17. Программные средства подбора паролей для ОС Windows 7/Vista/XP/2000/NT.
18. Средства защиты базы данных SAM в ОС Windows.
19. Реализация протокола Kerberos в ОС Windows.
20. Реализация протокола Kerberos в ОС Linux.
21. Атаки на асимметричные криптосистемы.
22. Атаки на механизм электронной цифровой подписи.
23. Разработка политики применения сертификатов.
24. Программные средства поддержки PKI..
25. Инструментальные программные средства реализации протокола SSL.
26. Обзор известных атак «человек посередине».
27. Защищенный вариант протокола RPC.
28. Аутентификация по «росписи» мышью.
29. Программное обеспечение смарт-карт и токенов.
30. Протокол CMP.
31. Аутентификация субъектов с использованием средств библиотеки Framework Class Library.
32. Аутентификация на основе RFID меток.
33. Аутентификация в беспроводных (Wi-Fi) сетях.
34. Средства разработки прикладных программ, использующих биометрическую аутентификацию пользователей.
35. Стандарты на запрос и получение сертификатов PKCS#10 и PKCS#7.
36. Другая тема, выбранная студентом и согласованная с преподавателем.

1. Критерии оценивания компетенций

Оценка «отлично» выставляется студенту, если он на высоком уровне: анализирует компьютерную систему с целью определения уровня защищенности и доверия; Применяет инструментальные средства проведения сертификационных испытаний; составляет и оформляет аналитический отчет по проведенным сертификационным испытаниям; делать выводы по оценке защищенности на основании аналитического отчета.

Оценка «хорошо» выставляется студенту, если на среднем уровне: анализирует компьютерную систему с целью определения уровня защищенности и доверия; Применяет инструментальные средства проведения сертификационных испытаний; составляет и оформляет аналитический отчет по проведенным сертификационным испытаниям; делать выводы по оценке защищенности на основании аналитического отчета.

Оценка «удовлетворительно» выставляется студенту, если на минимальном уровне: анализирует компьютерную систему с целью определения уровня защищенности и доверия; Применяет инструментальные средства проведения сертификационных испытаний; составляет и оформляет аналитический отчет по проведенным сертификационным испытаниям; делать выводы по оценке защищенности на основании аналитического отчета.

Оценка «неудовлетворительно» выставляется студенту, если он с грубыми ошибками: анализирует компьютерную систему с целью определения уровня защищенности и доверия; Применяет инструментальные средства проведения сертификационных испытаний; составляет и оформляет аналитический отчет по проведенным сертификационным испытаниям; делать выводы по оценке защищенности на основании аналитического отчета.

Оценка «отлично» выставляется студенту, если он на высоком уровне: разрабатывает программы и методики аттестационных испытаний объектов

вычислительной техники на соответствие требованиям по защите информации; проведения аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; подготовки заключения по результатам аттестации объектов вычислительной техники на соответствие требованиям по защите информации.

Оценка «хорошо» выставляется студенту, если на среднем уровне: разрабатывает программы и методики аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; проведения аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; подготовки заключения по результатам аттестации объектов вычислительной техники на соответствие требованиям по защите информации.

Оценка «удовлетворительно» выставляется студенту, если на минимальном уровне: разрабатывает программы и методики аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; проведения аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; подготовки заключения по результатам аттестации объектов вычислительной техники на соответствие требованиям по защите информации.

Оценка «неудовлетворительно» выставляется студенту, если он с грубыми ошибками: разрабатывает программы и методики аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; проведения аттестационных испытаний объектов вычислительной техники на соответствие требованиям по защите информации; подготовки заключения по результатам аттестации объектов вычислительной техники на соответствие требованиям по защите информации.

Оценка «отлично» выставляется студенту, если он на высоком уровне: проводит аттестационные испытания защищенности объектов вычислительной техники; проведения экспериментальных исследований уровней защищенности объектов вычислительной техники.

Оценка «хорошо» выставляется студенту, если на среднем уровне: проводит аттестационные испытания защищенности объектов вычислительной техники; проведения экспериментальных исследований уровней защищенности объектов вычислительной техники.

Оценка «удовлетворительно» выставляется студенту, если на минимальном уровне: проводит аттестационные испытания защищенности объектов вычислительной техники; проведения экспериментальных исследований уровней защищенности объектов вычислительной техники.

Оценка «неудовлетворительно» выставляется студенту, если он с грубыми ошибками: проводит аттестационные испытания защищенности объектов вычислительной техники; проведения экспериментальных исследований уровней защищенности объектов вычислительной техники.

2. Описание шкалы оценивания

Максимально возможный балл за весь текущий контроль устанавливается равным 55. Текущее контрольное мероприятие считается сданным, если студент получил за него не менее 60% от установленного для этого контроля максимального балла. Рейтинговый балл, выставляемый студенту за текущее контрольное мероприятие, сданное студентом в установленные графиком контрольных мероприятий сроки, определяется следующим образом:

Уровень выполнения контрольного задания	Рейтинговый балл (в % от максимального балла за контрольное задание)
Отличный	100
Хороший	80
Удовлетворительный	60
Неудовлетворительный	0

3. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

Процедура проведения данного оценочного мероприятия включает в себя:

- устный доклад по одной из указанных тем;

Предлагаемые студенту вопросы позволяют проверить ИД-2 ОПК-16, ИД-1 ПК-2, ИД-1 ПК-3, ИД-2 ПК-3 компетенции.

Для подготовки к данному оценочному мероприятию необходимо изучить рекомендованные источники и научную литературу и на их основании подготовиться к указанным в методических материалах вопросам для обсуждения. Для подготовки к данному оценочному мероприятию необходимо 15 часов, в течение которых обучающийся изучает научную и учебную литературу, составляет текст доклада и готовит презентацию.

При подготовке к ответу студенту предоставляется право пользования текстом доклада.

При проверке задания, оцениваются умение последовательно, четко и логически стройно излагать избранную проблему, обобщать, сопоставлять различные точки зрения по рассматриваемой проблеме, аргументировать основные положения и выводы, использовать научную литературу.

Оценочный лист студента (ки) _____ Ф.И.О.			
№ гр. _____			
Оценка складывается как среднее арифметическое из трех оценок: правильность ответа; умение приводить примеры; умение отвечать на дополнительные вопросы			
Оценка правильности ответа	Оценка умения приводить примеры	Оценка умения отвечать на дополнительные вопросы	Итоговая оценка

Список рекомендуемой литературы

Перечень основной литературы:

1. Лабораторный практикум по дисциплине Программно-аппаратные средства защиты информации Электронный ресурс / сост. И. А. Денисов. - Лабораторный практикум по дисциплине Программно-аппаратные средства защиты информации, 2022-04-04. - Москва : Московский технический университет связи и информатики, 2016. - 31 с. - Книга находится в премиум-версии ЭБС IPR BOOKS. - ISBN 2227-8397

2. Программно-аппаратные средства защиты информационных систем : учебное пособие / Ю.Ю. Громов, О.Г. Иванова, К.В. Стародубов, А.А. Кадыков ; Министерство образования и науки Российской Федерации ; Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Тамбовский государственный технический университет». - Тамбов : Издательство ФГБОУ ВПО «ТГТУ», 2017. - 194 с. : ил. - <http://biblioclub.ru/>. - Библиогр.: с. 190. - ISBN 978-5-8265-1737-6

Перечень дополнительной литературы:

1. Платонов, В. В. Программно-аппаратные средства защиты информации : учебник для вузов / В.В. Платонов. - Москва : Академия, 2013. - 330, [1] с. : ил. ; 22. - (Высшее профессиональное образование. Информационная безопасность. Бакалавриат). - Библиогр.: с. 326-327. - ISBN 978-5-7695-9327-7

Перечень учебно-методического обеспечения самостоятельной работы обучающихся по дисциплине (модулю)

Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины (модуля)

1. <http://elibrary.rsl.ru/> - Электронная библиотека РГБ.
2. <http://elibrary.ru/defaultx.asp> - Научная электронная библиотека
3. <http://www.intuit.ru> – Интернет-Университет Информационных Технологий.
4. <http://biblioclub.ru>