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

Методические указания

по выполнению лабораторных работ
по дисциплине
«Технологии и методы программирования»
для студентов
специальности 10.05.03 Информационная безопасность
автоматизированных систем
специализация Анализ безопасности информационных систем

Ставрополь
2026

ВВЕДЕНИЕ

Цель и задачи освоения дисциплины:

Целью освоения дисциплины «Технологии и методы программирования изучается» является формирование у будущего специалиста по специальности 10.05.03 Информационная безопасность автоматизированных систем специализация «Анализ безопасности информационных систем» понимания основных методов, средств и технологий эффективного взаимодействия и реализации успешной командной работы.

Задачи дисциплины:

- изучение теоретических основ формирования и развития навыков командной работы;
- формирование умений управления групповыми проектами;
- овладение навыками эффективного социального взаимодействия, создания благоприятной и конструктивной атмосферы в команде средствами доступных информационных технологий.

Семестр 3

Лабораторная работа 1. Жизненный цикл ПО. Модели жизненного цикла ПО

Теоретическая часть:

Жизненный цикл программного обеспечения представляет собой непрерывный процесс, начинающийся с момента возникновения идеи о создании ПО и заканчивающийся его полным выводом из эксплуатации, включая все промежуточные этапы разработки, сопровождения и модернизации. В индустрии программной инженерии сложились различные модели жизненного цикла, каждая из которых имеет свои преимущества и ограничения. Каскадная модель (Waterfall) характеризуется строгой последовательностью этапов: сбор и анализ требований, проектирование архитектуры, реализация, тестирование, внедрение и сопровождение, причем переход к следующему этапу возможен только после полного завершения предыдущего, что обеспечивает четкое планирование и контроль, но делает процесс недостаточно гибким для изменяющихся требований. Итеративная модель предполагает циклическую разработку с постепенным наращиванием функциональности, что позволяет получать промежуточные рабочие версии продукта и оперативно учитывать обратную связь от заказчиков. Гибкие методологии (Agile, Scrum, Kanban) основаны на принципах манифеста Agile и делают акцент на коротких итерациях (спринтах), постоянном взаимодействии с заказчиком, быстрой адаптации к изменениям требований и приоритете рабочего программного обеспечения перед comprehensive документацией. V-образная модель расширяет каскадный подход, тесно связывая этапы разработки с соответствующими уровнями тестирования, что особенно важно для критически важных систем с высокими требованиями к надежности. Спиральная модель Боэма сочетает элементы итеративного подхода с тщательным анализом рисков на каждом витке спирали, включая четыре основных квадранта: определение целей и ограничений, анализ рисков,

разработку и тестирование, планирование следующей итерации. Выбор конкретной модели зависит от множества факторов: характера и масштаба проекта, требований к качеству и надежности, сроков разработки, бюджета, уровня неопределенности требований, опыта команды и организационной культуры. В последние годы наблюдается тенденция к гибридизации подходов, когда формальные процессы сочетаются с гибкими практиками, что позволяет сохранить преимущества структурированного управления при сохранении необходимой гибкости. Особое внимание при выборе модели жизненного цикла уделяется вопросам документирования, управления изменениями требований, обеспечения качества на всех этапах разработки и взаимодействия между различными ролями в проекте.

Итоговые вопросы:

1. Каковы основные отличия между каскадной и итеративной моделями жизненного цикла ПО?

2. Почему гибкие методологии получили широкое распространение в современных проектах разработки ПО?

3. Какие факторы следует учитывать при выборе модели жизненного цикла для конкретного проекта?

Задания:

1. Сравните каскадную и Agile-модели для проекта разработки банковской системы.

2. Разработайте схему жизненного цикла для мобильного приложения с учетом возможных изменений требований.

3. Проанализируйте, какая модель жизненного цикла наиболее подходит для проекта с жесткими требованиями регуляторов.

Лабораторная работа 2. Стандарты в области разработки ПО (ISO, IEEE, SEI, ГОСТ Р)

Теоретическая часть:

Стандартизация в области разработки программного обеспечения играет crucial роль в обеспечении качества, надежности и безопасности создаваемых систем, а также в упорядочивании процессов разработки и взаимодействия между участниками проектов. Международные стандарты ISO/IEC 12207 "Системная и программная инженерия - Процессы жизненного цикла программных средств" и ISO/IEC 15288 "Системная инженерия - Процессы жизненного цикла систем" представляют собой всеобъемлющие framework, определяющие основные процессы и виды деятельности на протяжении всего жизненного цикла. Стандарт IEEE 829 регламентирует структуру и содержание тестовой документации, IEEE 1012 устанавливает требования к процессам верификации и валидации, IEEE 1028 определяет процедуры проведения инспекций и reviews. Институт Software Engineering Institute (SEI) при Carnegie Mellon University разработал модель зрелости процессов разработки CMMI (Capability Maturity Model Integration), которая позволяет организациям оценивать и совершенствовать свои процессы разработки. В российской практике широко применяются стандарты серии ГОСТ Р, включая ГОСТ Р 56939-2016 "Защита информации. Защита информации в системах разработки и производства программного обеспечения", ГОСТ 34 серии по автоматизированным системам управления, ГОСТ 19 серии по программной документации. Отраслевые стандарты, такие как DO-178C для авионики, IEC 62304 для медицинского ПО или ISO 26262 для автомобильной индустрии, устанавливают дополнительные требования для критически важных систем. Важность стандартов заключается в том, что они обеспечивают единый понятийный аппарат, формализуют best practices, помогают избежать типовых ошибок, облегчают взаимодействие между участниками проекта и обеспечивают соответствие регуляторным требованиям. Современные стандарты все чаще учитывают agile-подходы, что отражает эволюцию индустрии разработки ПО, однако сохраняют базовые принципы управления качеством и процессами.

Итоговые вопросы:

1. Какие основные группы стандартов существуют в области разработки ПО?

2. Как стандарты ISO/IEC 12207 и CMMI дополняют друг друга?

3. Почему отраслевые стандарты важны для специализированных областей?

Задания:

1. Сравните требования ГОСТ 34 и IEEE 829 к документации проекта.

2. Разработайте матрицу соответствия процессов вашей организации стандарту ISO/IEC 12207.

3. Проанализируйте, какие стандарты наиболее важны для проекта медицинского ПО.

Лабораторная работа 3. Процессы жизненного цикла ПО (стандарт ISO/IEC 12207)

Теоретическая часть:

Стандарт ISO/IEC 12207 "Информационные технологии. Процессы жизненного цикла программных средств" представляет собой всеобъемлющую framework для систематизации процессов разработки и сопровождения ПО, которая была впервые принята в 1995 году и с тех пор несколько раз обновлялась. В стандарте выделяются три основные группы процессов: основные процессы (acquisition, supply, development, operation, maintenance), вспомогательные процессы (documentation, configuration management, quality assurance, verification, validation, joint review, audit, problem resolution) и организационные процессы (management, infrastructure, improvement, training). Каждый процесс описывается через свои цели, ожидаемые результаты и ключевые виды деятельности, при этом стандарт не предписывает конкретных методов реализации, оставляя простор для

адаптации под нужды конкретной организации. Например, процесс разработки включает в себя анализ требований, архитектурное проектирование, детальное проектирование, кодирование, интеграцию компонентов, тестирование и установку готового решения. Процесс управления конфигурацией охватывает идентификацию конфигурационных единиц, контроль изменений, учет статуса и аудит конфигурации. Особенностью стандарта является его гибкость - организации могут адаптировать набор и детализацию процессов в зависимости от масштаба проекта, доменной области и применяемых методологий разработки. В последней редакции стандарта больше внимания уделено agile-практикам, управлению рисками, безопасности информации и непрерывному совершенствованию процессов. Реализация процессов по ISO/IEC 12207 помогает организациям систематизировать свою деятельность, улучшить качество продукции, повысить предсказуемость результатов и соответствовать требованиям регуляторов. Однако важно понимать, что стандарт задает what должно быть сделано, но не предписывает how это должно быть реализовано, что требует дополнительной адаптации под конкретные условия и компетенции команды.

Итоговые вопросы:

1. Какие три основные группы процессов определены в ISO/IEC 12207?

2. Почему стандарт не предписывает конкретных методов реализации процессов?

3. Как процессы верификации и валидации отличаются друг от друга?

Задания:

1. Сопоставьте процессы ISO/IEC 12207 с этапами вашего текущего проекта.

2. Разработайте схему взаимодействия основных и вспомогательных процессов.

3. Определите, какие процессы чаще всего игнорируются в небольших проектах и почему.

Лабораторная работа 4. Модель зрелости предприятия (CMM, CMMI)

Теоретическая часть:

Модель зрелости процессов разработки CMMI (Capability Maturity Model Integration) представляет собой эволюционное развитие более ранней модели CMM (Capability Maturity Model) и служит framework для оценки и совершенствования процессов в организации. Модель описывает пять уровней зрелости процессов: Initial (процессы выполняются ad-hoc, успех зависит от отдельных людей), Managed (процессы характеризуются для отдельных проектов, есть базовое управление), Defined (процессы стандартизированы и документированы для всей организации), Quantitatively Managed (процессы измеряются и контролируются с использованием метрик), Optimizing (фокус на постоянном улучшении процессов на основе количественного анализа). Каждый уровень включает несколько ключевых областей процессов (Process Areas), таких как Requirements Management, Project Planning, Process and Product Quality Assurance для второго уровня. Реализация CMMI требует значительных организационных изменений, но дает существенные преимущества: предсказуемость сроков и бюджета проектов, повышение качества продукции, снижение рисков, лучшую управляемость распределенными командами. В последних версиях CMMI (версия 2.0) появилась возможность выбора представления модели - staged (по уровням) или continuous (по процессным областям), что повышает гибкость ее применения для различных типов организаций. Оценка по CMMI проводится сертифицированными lead appraisers и может служить важным конкурентным преимуществом при участии в тендерах, особенно для компаний, работающих с государственными заказчиками или в регулируемых отраслях, таких как

оборонная промышленность, здравоохранение или финансовый сектор. Важно отметить, что внедрение CMMI - это длительный процесс организационной трансформации, который требует вовлечения руководства, изменения корпоративной культуры и постоянного мониторинга показателей процессов.

Итоговые вопросы:

1. Какие основные уровни зрелости определены в модели CMMI?

2. Почему переход между уровнями зрелости требует организационных изменений?

3. Как модель CMMI помогает в управлении крупными распределенными проектами?

Задания:

1. Проведите самооценку процессов вашей организации по модели CMMI.

2. Разработайте план перехода с текущего уровня зрелости на следующий.

3. Сравните staged и continuous представления модели CMMI.

Лабораторная работа 5. Основные этапы разработки сложных программных систем

Теоретическая часть:

Разработка сложных программных систем представляет собой многоэтапный процесс, который требует тщательного планирования и управления на протяжении всего жизненного цикла. На этапе инициации проекта проводится анализ бизнес-потребностей, оценка осуществимости (feasibility study), определение стейкхолдеров и их требований, формирование устава проекта. Этап сбора и анализа требований включает выявление функциональных требований (что должна делать система), нефункциональных

требований (качество, производительность, безопасность), их приоритезацию и документирование в виде спецификаций требований (SRS). Проектирование системы охватывает создание архитектурных решений (high-level design), выбор технологического стека, проектирование компонентов и интерфейсов (detailed design), разработку моделей данных, определение стратегии интеграции. Непосредственная реализация (кодирование) предполагает написание исходного кода в соответствии с принятыми стандартами кодирования, best practices и принципами clean code. Этап тестирования включает модульное тестирование (проверка отдельных компонентов), интеграционное тестирование (проверка взаимодействия компонентов), системное тестирование (проверка системы в целом) и приемочное тестирование (валидация соответствия требованиям заказчика). Развертывание системы может проводиться по различным схемам: Big Bang (полный переход за один раз), поэтапное внедрение (по функциональности или подразделениям) или параллельный запуск (старая и новая системы работают одновременно). Эксплуатация и сопровождение включают мониторинг работы системы, устранение инцидентов, внесение изменений и обновлений, оптимизацию производительности. Особенностью сложных систем является необходимость управления конфигурацией (версиями компонентов), интеграцией (сборкой системы из отдельных частей), координацией работы распределенных команд и управления зависимостями между компонентами. Современные подходы к разработке сложных систем все чаще используют принципы DevOps и непрерывной интеграции/доставки (CI/CD) для ускорения вывода изменений в production при сохранении необходимого уровня качества и стабильности системы.

Итоговые вопросы:

1. Какие виды тестирования применяются на разных этапах разработки?

2. Почему проектирование архитектуры критически важно для сложных систем?

3. Как принципы DevOps меняют традиционные этапы разработки?

Задания:

1. Разработайте план этапов для системы электронного документооборота.

2. Сравните стратегии развертывания для критически важной системы.

3. Определите ключевые риски на каждом этапе жизненного цикла.

Лабораторная работа 6. Структура и состав технического задания в соответствии с ГОСТ

Теоретическая часть:

Техническое задание (ТЗ) является основным документом, определяющим требования к разрабатываемой системе и служащим основанием для проведения работ. В российской практике структура технического задания регламентируется ГОСТ 34.602-89 "Техническое задание на создание автоматизированной системы" и включает следующие основные разделы: общие положения (основание для разработки, назначение системы, краткая характеристика объекта автоматизации), технические требования к системе (требования к структуре и функционированию системы, требования к надежности, безопасности, эргономике), состав и содержание работ (этапы разработки, сроки выполнения), порядок контроля и приемки (виды испытаний, процедура приемки), требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие. Особое внимание в ТЗ уделяется формулировке требований - они должны быть конкретными, измеримыми, достижимыми, релевантными и ограниченными по времени (SMART-критерии). Для сложных систем техническое задание может включать приложения: схемы бизнес-процессов, прототипы интерфейсов, глоссарий терминов. В процессе разработки ТЗ важно обеспечить участие всех ключевых стейкхолдеров: заказчиков,

пользователей, разработчиков, экспертов предметной области. Качественно составленное техническое задание позволяет минимизировать риски недопонимания между заказчиком и исполнителем, служит основой для планирования работ и оценки их результатов, является документом, на который можно ссылаться при возникновении спорных ситуаций. В agile-подходах роль технического задания частично выполняет product backlog, однако даже в этом случае важно документировать ключевые архитектурные решения и нефункциональные требования.

Итоговые вопросы:

1. Какие основные разделы включает ТЗ по ГОСТ 34.602-89?
2. Почему формулировка требований должна соответствовать SMART-критериям?
3. Как изменилась роль ТЗ при переходе к agile-методологиям?

Задания:

1. Разработайте техническое задание для системы учета заказов.
2. Проведите анализ качества требований в предоставленном ТЗ.
3. Сравните структуру ТЗ по ГОСТ и product backlog в Scrum.

Лабораторная работа 7. Этапы разработки ПО в соответствии с ГОСТ

Теоретическая часть:

Разработка программного обеспечения по ГОСТу (например, ГОСТ 19 и ГОСТ 34) регламентирует структуру проектной документации и последовательность этапов жизненного цикла. Согласно ГОСТ 34.601-90, жизненный цикл автоматизированной системы (включая ПО) включает следующие основные этапы: формирование требований, разработка концепции, техническое задание, эскизный проект, технический проект, рабочая документация, внедрение, сопровождение. Каждый этап завершается

выпуском документа, соответствующего ГОСТу, например, Техническое задание (ТЗ), Технический проект (ТП), Рабочая документация (РД). ГОСТ обеспечивает формализацию процессов, единообразие подходов и гарантирует прослеживаемость требований на всех этапах.

Итоговые вопросы:

1. Какие этапы жизненного цикла ПО определяются ГОСТ 34.601-90?
2. Чем отличаются документы «Техническое задание» и «Технический проект»?
3. Как ГОСТ помогает в обеспечении качества и управляемости разработки?

Задания:

1. Составьте структуру документа «Техническое задание» по ГОСТ.
2. Опишите этап «Технический проект» для системы управления складом.
3. Проанализируйте преимущества и недостатки ГОСТ-подхода по сравнению с гибкими методологиями.

Лабораторная работа 8. Методологии разработки ПО: РУП, МСФ, ХР

Теоретическая часть:

Методологии разработки программного обеспечения определяют организацию процесса создания ПО. RUP (Rational Unified Process) — итеративная методология, разделённая на фазы (инициация, разработка, реализация, сопровождение) и роли. MSF (Microsoft Solutions Framework) предлагает гибкий фреймворк, основанный на командной модели, управлении рисками и итеративной разработке. XP (Extreme Programming) — это гибкая методология, акцентирующая внимание на частых релизах, парном программировании, постоянной интеграции и тесном взаимодействии с заказчиком. Все три подхода стремятся повысить управляемость, качество и

адаптивность проекта, но имеют разный уровень формальности и применяются в различных контекстах.

Итоговые вопросы:

1. В чем принципиальные отличия RUP, MSF и XP?
2. Какие практики XP направлены на улучшение качества кода?
3. Как выбор методологии влияет на структуру команды и планирование?

Задания:

1. Сравните RUP и XP по структуре и применимости.
2. Опишите этапы проекта в рамках MSF.
3. Выберите подходящую методологию для стартапа и обоснуйте выбор.

9. Показатели качества ПО. Стандарт ИСО/МЭК 9126

Теоретическая часть:

Стандарт ISO/IEC 9126 описывает модель оценки качества программного обеспечения. Она включает шесть основных характеристик: функциональность (наличие необходимых функций), надежность (устойчивость к ошибкам), удобство использования (usability), эффективность (ресурсоемкость и производительность), сопровождаемость (простота модификаций), портируемость (переносимость на другие платформы). Каждая характеристика включает подхарактеристики, например, надежность включает устойчивость, восстановимость и зрелость. Этот стандарт позволяет формализованно оценивать и сравнивать ПО, выявлять слабые стороны и управлять качеством на всех этапах жизненного цикла.

Итоговые вопросы:

1. Какие характеристики включает модель качества ISO/IEC 9126?
2. Почему сопровождаемость важна в длительных проектах?
3. Как измеряется эффективность программного обеспечения?

Задания:

1. Приведите примеры показателей для каждой характеристики ISO/IEC 9126.

2. Оцените качество существующей системы по этой модели.

3. Предложите метрики для оценки «удобства использования» и «надежности».

Семестр 4

Лабораторная работа 1. Основы планирования разработки программного обеспечения. Подходы к сложности и времени разработки ПО

Теоретическая часть:

Планирование разработки ПО включает определение объема работ, временных рамок, ресурсов, бюджета и рисков. Основные задачи — постановка целей, декомпозиция задач, оценка трудозатрат и построение графика проекта (например, с помощью диаграммы Ганта). Сложность проекта может оцениваться с учетом количества функций, интеграций, технологий, команды. Время разработки зависит от объема задач и доступных ресурсов, и может оцениваться с помощью экспертных методов, аналогий или метрик (например, LOC, FP).

Итоговые вопросы:

- Какие задачи включает этап планирования проекта?
- От каких факторов зависит сложность разработки?
- Как оценивается длительность проекта?

Задания:

- Составьте план-график разработки небольшого проекта.
- Определите ключевые ресурсы для команды из 5 человек.
- Оцените сложность проекта по числу функций и интеграций.

Лабораторная работа 2. Размерно- и функционально-ориентированные метрики: LOC, FP

Теоретическая часть:

Метрики LOC (Lines of Code) и FP (Function Points) применяются для оценки размера и сложности ПО. LOC — простая количественная метрика, определяющая объем кода. FP — функционально-ориентированная метрика, оценивающая систему по числу входов, выходов, запросов и файлов. FP независим от языка программирования и подходит для ранних этапов оценки. Метрики помогают планировать ресурсы, трудозатраты, и оценивать производительность команды.

Итоговые вопросы:

- В чем разница между LOC и FP?
- Какие преимущества у FP перед LOC?
- Почему важно оценивать размер ПО?

Задания:

- Подсчитайте LOC для выбранного модуля.
- Определите количество FP по описанию требований.
- Сравните LOC и FP на примере одного проекта.

Лабораторная работа 3. Принципы разработки и организация коллективов разработчиков ПО

Теоретическая часть:

Организация команды разработчиков играет ключевую роль в успехе проекта. Возможны различные подходы: иерархическая модель (архитектор — ведущие разработчики — программисты), скрам-команды, кросс-функциональные группы. Важно обеспечить распределение ролей, координацию, коммуникацию и контроль задач. Принципы разработки — модульность, повторное использование, расширяемость, сопровождаемость — лежат в основе архитектурных решений.

Итоговые вопросы:

- Какие существуют модели организации команд?
- В чем преимущества кросс-функциональных групп?
- Какие принципы обеспечивают качество архитектуры?

Задания:

- Нарисуйте структуру команды для веб-проекта.
- Распределите роли и задачи в мини-группе.
- Опишите процесс взаимодействия команды в Agile.

Лабораторная работа 4. Инструменты управления проектами разработки ПО (JIRA, Trello, Asana)

Теоретическая часть:

Инструменты управления проектами позволяют отслеживать задачи, прогресс и коммуникацию команды. **JIRA** — мощный инструмент для гибкой разработки, с поддержкой Scrum и Kanban. **Trello** — визуальная доска с карточками, удобна для небольших команд. **Asana** — ориентирована на задачи и коллаборацию, с гибкой структурой. Все эти системы позволяют распределять задачи, назначать исполнителей, отслеживать сроки и статусы.

Итоговые вопросы:

- Какие функции реализуют JIRA, Trello и Asana?
- В чем преимущества Kanban-досок?
- Как инструменты помогают управлять дедлайнами?

Задания:

- Создайте Kanban-доску в Trello для проекта.
- Запланируйте спринт в JIRA.
- Сравните интерфейс и функции JIRA и Asana.

Лабораторная работа 5. Методы тестирования ПО: модульное, интеграционное, системное, приемочное

Теоретическая часть:

Тестирование — ключевой этап контроля качества ПО. **Модульное тестирование** проверяет отдельные функции/модули. **Интеграционное** — взаимодействие между модулями. **Системное тестирование** — проверка всей системы целиком. **Приемочное тестирование** — валидация требований

заказчика. Разные уровни тестирования позволяют выявить ошибки на разных этапах и минимизировать риски на продакшене.

Итоговые вопросы:

- Чем отличается интеграционное тестирование от модульного?
- Когда проводится системное тестирование?
- Кто выполняет приемочное тестирование?

Задания:

- Приведите пример модульного теста.
- Составьте план интеграционного тестирования.
- Опишите приемочное тестирование интернет-магазина.

Лабораторная работа 6. Автоматизация внедрения: инструменты и подходы

Теоретическая часть:

Автоматизация внедрения позволяет упростить и ускорить выпуск новых версий ПО. Используются инструменты CI/CD, скрипты деплоя, конфигурационные менеджеры. Подходы: blue-green deployment, rolling update, canary deployment. Инструменты: Jenkins, GitLab CI/CD, Ansible, Docker, Kubernetes. Автоматизация снижает человеческий фактор и повышает стабильность релизов.

Итоговые вопросы:

- Что такое blue-green deployment?
- Какие инструменты используются для деплоя?
- Как автоматизация повышает надежность внедрения?

Задания:

- Опишите процесс автоматического деплоя через GitLab CI.
- Настройте простую задачу в Jenkins.
- Сравните canary и rolling deployment.

Лабораторная работа 7. Контроль версий и системы управления

версиями (Git, SVN)

Теоретическая часть:

Системы управления версиями (VCS) позволяют отслеживать изменения в коде, работать в команде, восстанавливаться к предыдущим версиям. **Git** — распределенная VCS, популярная благодаря гибкости и интеграции с GitHub, GitLab. **SVN** — централизованная система, подходит для больших корпоративных репозиториев. Контроль версий обеспечивает прозрачность и защиту от потерь.

Итоговые вопросы:

- Чем отличается распределенная и централизованная VCS?
- Как работают ветки в Git?
- Какие команды используются для коммита и слияния?

Задания:

- Выполните базовые команды Git: init, add, commit, push.
- Создайте и слейте ветку.
- Сравните Git и SVN по архитектуре.

Лабораторная работа 8. Непрерывная интеграция и непрерывное развертывание (CI/CD)

Теоретическая часть:

CI/CD — практика автоматизации всех этапов от написания кода до выпуска в production. **CI** (Continuous Integration) — автоматическое тестирование и сборка при каждом коммите. **CD** (Continuous Delivery/Deployment) — автоматическая доставка и развертывание. Преимущества: быстрая доставка изменений, контроль качества, сокращение рисков. Используются инструменты Jenkins, GitHub Actions, GitLab CI.

Итоговые вопросы:

- Что отличает CI от CD?
- Какие этапы включает CI/CD pipeline?
- Какие инструменты CI/CD вам известны?

Задания:

- Опишите пайплайн CI для веб-приложения.
- Настройте автосборку и автотесты.
- Сравните Jenkins и GitHub Actions.

Лабораторная работа 9. DevOps: принципы и практика

Теоретическая часть:

DevOps объединяет разработку (Dev) и эксплуатацию (Ops), создавая культуру сотрудничества, автоматизации и быстрой доставки. Принципы: автоматизация, мониторинг, CI/CD, инфраструктура как код. DevOps-культура снижает барьеры между командами, повышает стабильность и позволяет быстрее реагировать на изменения. Используются инструменты Docker, Kubernetes, Terraform, Prometheus.

Итоговые вопросы:

- Какие задачи решает DevOps?
- Что такое инфраструктура как код?
- Как DevOps связан с CI/CD?

Задания:

- Нарисуйте DevOps-процесс для небольшого проекта.
- Опишите роль DevOps-инженера.
- Настройте CI/CD пайплайн с Docker.

Лабораторная работа 10. Безопасность ПО: уязвимости, методы защиты, стандарты безопасности

Теоретическая часть:

Безопасность ПО включает защиту от внешних и внутренних угроз: XSS, SQL-инъекций, CSRF, уязвимостей авторизации. Методы: безопасное программирование, аудит кода, тесты на проникновение. Стандарты: OWASP Top 10, ISO/IEC 27001. Реализация защиты — часть SDLC (жизненного цикла разработки ПО), включающая шифрование, контроль доступа, логирование и

мониторинг.

Итоговые вопросы:

- Какие наиболее распространенные уязвимости существуют?
- Как OWASP помогает разработчикам?
- Почему важно учитывать безопасность на всех этапах?

Задания:

- Проанализируйте уязвимости простого веб-приложения.
- Опишите меры защиты от XSS и SQL-инъекций.
- Составьте чек-лист безопасной разработки.

Итоговый тест

1. Что означает принцип DRY в программировании?

A) Do Repeat Yourself

B) Don't Repeat Yourself (правильный ответ)

C) Debug and Rewrite Yourself

D) Data Replacement Yield

2. Какой из перечисленных языков программирования является строго типизированным?

A) Python

B) JavaScript

C) Java (правильный ответ)

D) PHP

3. Что такое инкапсуляция в объектно-ориентированном программировании?

A) Наследование методов от родительского класса

B) Создание новых классов

C) Сокрытие внутренних деталей реализации объекта (правильный ответ)

D) Использование шаблонов проектирования

4. Какой метод сортировки имеет наихудшую временную сложность \(O(n^2) \)?

A) Merge Sort

B) Quick Sort

C) Bubble Sort (правильный ответ)

D) Heap Sort

5. Что такое Git?

A) Компилятор C++

B) Язык программирования

C) Система управления базами данных

D) Система контроля версий (правильный ответ)

6. Какой оператор используется в C++ для динамического выделения памяти?

A) malloc

B) allocate

C) new (правильный ответ)

D) memory

7. Что такое рекурсия?

A) Процесс обработки исключений

B) Способ организации баз данных

C) Функция, вызывающая саму себя (правильный ответ)

D) Способ передачи параметров

8. Какой паттерн проектирования относится к порождающим?

A) Singleton (правильный ответ)

B) Observer

C) Adapter

D) Proxy

9. Что означает принцип SOLID в программировании?

A) Совокупность пяти принципов написания читаемого и масштабируемого кода (правильный ответ)

B) Метод оптимизации памяти

C) Алгоритм шифрования данных

D) Фреймворк для быстрой разработки

10. Какой результат работы следующего кода на Python: `print("1" + "2")`?

A) 3

B) 12 (правильный ответ)

C) Ошибка

D) None

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Методические указания

по выполнению курсового проекта по дисциплине

«**Технологии и методы программирования**»

для студентов специальности

10.05.03 Информационная безопасность автоматизированных систем

Введение

Настоящие рекомендации предназначены для студентов всех форм обучения при выполнении курсовых работ, для руководителей курсовых работ с целью формирования единых требований при разработке и защите работ. Рекомендации содержат предложения по структуре курсовой работы (далее - работа), объему, содержанию, оформлению, процедуре допуска к защите и порядку защиты работ.

Целями выполнения курсовой работы являются:

- обогащение знаний студентов,
- обучение методам теоретического анализа явлений и закономерностей науки,
- выработка навыков применения теоретических знаний к комплексному решению профессиональных задач, использования справочной литературы, методов математической обработки экспериментальных данных, компьютерных технологий.

Системой курсовых работ (проектов) студент подготавливается к выполнению выпускной квалификационной работы.

Перед прочтением Методических указаний ознакомьтесь с основными ошибками, которые допускают студенты, приступая к написанию курсовой работы. Возможно, это мотивирует на более внимательное отношение к требованиям по написанию курсовой работы.

1. Выбрав тему, студент выходит в сеть Интернет и через поисковую систему (Yandex, Google и т.д. – зависит от предпочтений) находит курсовую работу по данной теме, меняет реквизиты, Ф.И.О. автора и сдаёт преподавателю. В результате получает рецензию: «Плагиат. На переработку».
2. Аналогичные действия п.1, но студент подходит к решению проблемы творчески и находит несколько курсовых работ по данной теме. Компонует материал по своему усмотрению и сдает преподавателю под своим авторством. В результате получает рецензию: «Плагиат. На переработку». Почему дана такая рецензия? По трем основным причинам: во-первых, написание курсовых работ – распространённый бизнес в Интернете и качественный материал бесплатно не выкладывают в сети, таким образом, компоновать приходится из «третьесортного сырья»; во-вторых, курсовая уже написана и защищена ранее, то есть она уже устарела и не отражает современных реалий, в-третьих, такие работы редко соответствуют требованиям, предъявляемым кафедрой к курсовым работам.
3. Студент принимает постороннюю помощь в написании работы. Здесь есть несколько аспектов, но следует обратить внимание на два основных. Во-первых, помощь исходит, как правило, от одного физического или юридического лица для группы студентов. В результате преподаватель получает на проверку несколько работ на разные темы, написанные «одной рукой». Формальных причин для отрицательной рецензии у преподавателя нет, особенно если работа выполнена в строгом соответствии с требованиями, но, как показывает практика, защита таких работ проходит безуспешно. Во-вторых, студент, не приложивший усилий к данной работе, как правило, очень плохо в ней ориентируется, особенно, когда преподаватель уточняет аспекты рассматриваемой темы. Но так как Вы уже вышли на защиту, то получаете заслуженную оценку «неудовлетворительно».
4. Вы самостоятельно осуществляете работу, но так же самостоятельно выбираете требования к курсовой работе. Если Вам они нравятся и понятны, то исполняете, а если не нравятся, пропускаете. Это очень распространённая ошибка. Выполнять надо все без исключения требования к курсовой работе. Если в процессе работы возникли вопросы, следует по ним получить консультацию у научного руководителя.

Это краткий перечень основных ошибок, которые недопустимо совершать после прочтения данных Методических указаний.

1. ОРГАНИЗАЦИЯ ВЫПОЛНЕНИЯ И ЗАЩИТЫ КУРСОВОЙ РАБОТЫ

Виды работ по этапам выполнения курсовой работы:

а) Предварительный этап:

- выбор темы курсовой работы и оценка возможности раскрытия данной темы;
- подача заявления на закрепление темы;

б) Основной этап:

- сбор материала;
- оформление курсовой работы;

в) Заключительный этап:

- рецензирование работы руководителем и допуск к защите;
- защита работы.

1.1 Предварительный этап

Темы курсовых работ определяются выпускающей кафедрой. Студенту предоставлено право выбора темы курсовой работы. Задание к выбранной теме выдается руководителем работы (проекта) и регистрируется в кафедральном журнале.

1.2 Основной этап

Составление плана курсовой работы

План курсовой работы представляет собой составленный в определенном порядке перечень глав и развернутый перечень вопросов, которые должны быть освещены. Правильно составленный план работы помогает систематизировать материал, обеспечивает последовательность его изложения. План работы составляется самостоятельно, с учетом замысла и индивидуального подхода. План работы рекомендуется **согласовать с руководителем курсовой работы**. В процессе работы план работы может уточняться.

Подбор литературы

Курсовая работа выполняется на основе глубокого изучения литературных источников. Литература по теме работы может быть подобрана студентом при помощи предметных и алфавитных каталогов библиотек. Для этих целей могут быть использованы каталоги книг, указатели журнальных статей, специальные библиографические справочники, тематические сборники литературы, периодически выпускаемые отдельными издательствами, интернет-сайты официальных организаций.

Проработка источников сопровождается выписками, конспектированием. Выписки из текста делают обычно в виде цитаты. После каждой цитаты приводится ссылка на автора и источник. Поэтому при выписке цитат и конспектировании следует делать ссылки: автор, название издания, место издания, издательство, год издания, номер страницы. Конспект может содержать в себе факты, доказательства, примеры, основные положения и выводы автора, а также личное отношение студента к изученному материалу.

Рациональной и эффективной организации исследовательской деятельности способствует составление студентом собственной картотеки проработанных по теме и смежным вопросам литературных источников. Картотека не только помогает получить представление о степени изученности данной проблемы, но и позволяет работать с литературой более целеустремленно.

Оформление курсовой работы

Оформление проекта или работы осуществляется в соответствии с требованиями положения о выполнении и защите курсовых о выполнении и защите курсовых работ (проектов).

Содержание и структура курсовых работ

Курсовая работа должна содержать элементы новизны, наряду с фундаментальным аспектом должен быть проведен анализ современного состояния изучаемой проблемы, а также отражены региональные особенности. Задание по курсовой работе необходимо индивидуализировать с учетом интересов и способностей студентов.

Курсовая работа должна состоять из введения, теоретической части, эмпирической (практической, расчетно-графической) части, заключения, списка литературы и приложения. В отдельных случаях, в соответствии с тематикой работы, эмпирическая часть может отсутствовать.

Во введении обосновывается актуальность выбранной темы исследования, задачи, цели, новизна, теоретическая и практическая значимость исследования.

Теоретическая часть должна содержать анализ состояния изучаемой проблемы на основе обзора научной, научно-информационной, справочной литературы. Представленный материал должен быть логически связан с целью исследования. В параграфах теоретической части необходимо отражать отдельные компоненты проблемы и завершать их выводами.

Эмпирическая (практическая, расчетно-графическая) часть (при наличии) включает описание системы экспериментального исследования, обоснование методов исследования, анализ результатов экспериментального исследования, схемы, графические и математические способы интерпретации полученных данных, выводы.

Заключение содержит выводы, подтверждающие или опровергающие первоначальные предположения (гипотезы), перспективы дальнейшего изучения проблемы, связь с практикой, анализ реализации целей и задач исследования.

Список литературы должен быть составлен в соответствии с требованиями к оформлению библиографий.

Приложение содержит весь фактический материал экспериментальных исследований (анкеты, опросники, схемы, чертежи, расчетные материалы, карты, рисунки, ответы респондентов и т.д.).

Содержание курсовой работы (проекта) рекомендуется представлять в объеме 20-30 страниц печатного текста. Текст работы должен быть напечатан через 1,5 интервала на одной стороне стандартного листа белой бумаги (А - 4). Текст и другие отпечатанные элементы работы должны быть черными, контуры букв и знаков – четкими, без ореола и затенения. Шрифт TimesNewRoman, кегель 14. Курсив и подчеркивание в работе не допускаются. Названия глав и параграфов выделяются полужирным шрифтом. Лист с текстом должен иметь поля: слева – 30 мм, справа – 10 мм, сверху – 20 мм, снизу – 20 мм. Нумерация страниц текста делается в правом нижнем углу листа. Проставлять номер страницы необходимо со страницы, где печатается "Введение", на которой ставится цифра "3". После этого нумеруются все страницы, включая приложения.

Между названием главы и названием параграфа этой главы ставится пробел равный двум интервалам, а название параграфа не должно отделяться от текста этого параграфа пробелом. Названия параграфов отделяются от текста предыдущего параграфа пробелом, равным двум интервалам. Каждая глава, а также введение, выводы, предложения и список использованной литературы начинаются с новой страницы. Слово "Глава" не пишется. Главы имеют порядковые номера в пределах всей работы, обозначаемые арабскими цифрами (например: 1,2,3), после которых ставится точка. Слово "параграф" или значок параграфа в названии не ставятся. Параграфы имеют порядковые номера в пределах глав, обозначаемые арабскими цифрами (например: 1.1. и 1.2.). Заголовки глав и параграфов в тексте работы должны располагаться по центру, точку в конце названия главы и параграфа не ставят. Не допускается переносить часть слова в заголовке.

Нумерация таблиц и рисунков может быть сквозной или соотноситься с номером главы и параграфа. Например, если таблица или рисунок включены в текст первого параграфа второй главы, нумерация следующая: Таблица 2.1.1., рис. 2.1.1. Последняя цифра означает порядковый номер таблицы (или рисунка) в данном параграфе. Таблица помещается в качестве следующей страницы после первого упоминания о ней в тексте.

В рамках отведенного для руководства курсовыми работами времени регулярно проводятся индивидуальные консультации. Руководитель работы осуществляет предварительный просмотр выполненных заданий. После проверки выполнения студентом одного этапа работы, руководитель работы разрешает студенту перейти к следующему.

1.3 Заключительный этап

Работа сдается руководителю. Если работа удовлетворяет требованиям, предъявляемым к курсовым работам, она допускается к защите.

Защита курсовой работы является обязательной формой проверки выполнения работы. Защита производится на заседании кафедры, научно-методического семинара кафедры, научной проблемной группы при непосредственном участии руководителя, в присутствии студентов. Результаты наиболее интересных курсовых работ могут быть доложены на научных конференциях.

Защита состоит в коротком докладе студента по выполненной работе и в ответах на вопросы присутствующих на защите.

Результаты защиты курсовой работы, согласно действующему Положению о текущем контроле и промежуточной аттестации, оцениваются дифференцированной отметкой по пятибалльной системе.

Студент, не представивший в установленный срок работу, получивший неудовлетворительную оценку на защите или не явившийся на защиту по неуважительной причине, считается имеющим академическую задолженность.

Для ликвидации академической задолженности студент обязан в установленные сроки представить курсовую работу руководителю и защитить её перед комиссией.

Допуск к защите курсовой работы

Допуск к защите осуществляет научный руководитель курсовой работы.

Работа не допускается к защите, если она не носит самостоятельного характера, списана из литературных источников или у других авторов, если основные вопросы не раскрыты, изложены схематично, фрагментарно, в тексте содержатся ошибки, научный аппарат оформлен неправильно, текст написан небрежно.

Неудовлетворительно выполненная работа подлежит переработке в соответствии с

замечаниями преподавателя, содержащимися в отзыве.

При получении допуска к защите студент вправе получить дополнительную консультацию по предстоящей защите.

Курсовая работа, студенту не возвращается и хранится на кафедре в течение 2-х лет.

После получения допуска к защите, студент самостоятельно готовится к защите: составляет текст доклада, реагирует на замечания руководителя.

Подготовка к защите курсовой работы включает устранение ошибок и недостатков, изучение дополнительных источников, готовность объяснить любые приведенные в работе положения.

В ходе защиты курсовой работы задача студента - показать углубленное понимание вопросов конкретной темы, хорошее владение материалом по теме.

Защита работы производится в соответствии с планом приёма защит курсовых работ. С ним можно ознакомиться на стенде кафедры или непосредственно уточнить у научного руководителя время и место защиты.

По результатам защиты курсовой работы в зачётную ведомость (для защиты курсовой работы) выставляется оценка.

Критерии оценки курсовой работы:

- «**отлично**» выставляется студенту, показавшему глубокие знания, которые применяются при самостоятельном исследовании избранной темы, умение обобщать практический материал и делать на основе анализа выводы. Курсовая работа с оценкой «отлично» может быть рекомендована на конкурс студенческих работ, использована в качестве доклада на научно-студенческой конференции.
- «**хорошо**» выставляется студенту, показавшему в работе и при её защите полное знание материала, всесторонне осветившему вопросы темы, но не полной мере проявившему самостоятельность в исследовании.
- **«удовлетворительно»** выставляется студенту, раскрывшему в работе основные вопросы избранной темы, но не проявившему самостоятельности в анализе или допустившему отдельные неточности содержания работы.
- «**неудовлетворительно»** выставляется студенту, не раскрывшему основные положения избранной темы и допустившему грубые ошибки в содержании работы.

2. СОДЕРЖАНИЕ КУРСОВОЙ РАБОТЫ

2.1 Структурные элементы курсовой работы

Структурными элементами курсовой работы являются:

1) титульный лист,
2) оглавление,
5) введение;
6) основная часть;
7) заключение;
8) список использованных источников;
9) приложения.

Титульный лист является первой страницей курсовой работы.

Оглавление включает введение, наименование всех глав и параграфов, заключение, список использованных источников и приложения с указанием номеров страниц, с которых начинаются эти элементы работы. Следует обратить внимание, что через оглавление прослеживается структура всей работы. Желательно основную часть работы уместить в 2 главы, каждая из которых разбивается на 2-3 параграфа. Глава не может содержать один параграф. Наличие в главе более трех параграфов говорит о поверхностном представлении данных вопросов, так как к работе предъявляются жесткие требования по объему работы.

Введение. Во введении обосновывается **актуальность темы**, **формулируются цель** курсовой работы **и комплекс взаимосвязанных задач**, подлежащих решению в процессе работы, а также представляется **теоретическая и методологическая база.**

Актуальность темы отражает степень важности её раскрытия на данный момент, то есть, почему читателю будет интересно, важно или просто необходимо познакомиться с Вашей работой. Это 2-3 предложения, которые анонсируют Вашу работу. Важно во введении**,** раскрывая актуальность темы, не уйти в раскрытие заявленной проблемы. Не следует писать фразы типа:

«Мне эта тема очень интересна, поэтому она актуальна». Здесь необходимо заинтересовать читателя предметно и содержательно, а не своим примером.

Цель работы – это результат, к которому Вы желаете прийти и путь к нему. Цель работы одна. Формулируем её с использованием неопределённой формы глагола: изучить, исследовать, проанализировать, рассмотреть и т.д. Цель вашей курсовой работы скрыта в названии работы.

Задачи работы отражают шаги, делая которые, Вы достигаете цели. Вы должны поставить 6 - 7 задач. Они фактически связаны с параграфами Вашей работы. Задачи формулируются аналогично цели с использованием неопределенной формы глагола: изучить, исследовать, проанализировать, рассмотреть, оценить, выявить и т.д.

Теоретическая и методологическая база подразумевает краткое описание использованных источников с указанием основных авторов. Это «поклон» в адрес людей, чьими трудами Вы воспользовались, а также возможность для специалиста с первого взгляда понять, о чем написана работа и какого подхода к данной проблеме придерживаетесь Вы.

Объем этой части работы 1 лист.

Так как введение – это та часть работы, которая подразумевает 100% самостоятельное изложение, важно не перейти на «ты» в общении с подразумеваемым читателем, то есть **всё изложение материала должно осуществляется в неопределенной форме глагола**. Нельзя употреблять местоимение «я».

Основная часть. Требования к содержанию основной части работы даны ранее.

Заключение. В заключении формулируются краткие выводы, полученные в ходе выполнения поставленных задач и цели. Вы кратко излагаете то, что написали в работе, но это самое главное, что вы хотите донести до читателя, самый «сок» Вашего труда. В заключение можно вынести числовые данные в целях иллюстрации вывода. Они должны отражать цель и задачи исследования, и не быть «вырванными» из работы числами.

Список использованных источников должен содержать сведения об источниках, использованных при выполнении работы.

Приложения. В приложении рекомендуется включать материалы, связанные с выполненной работой, которые по каким-либо признакам не могут быть включены в основную часть.

2.2 Содержание основной части курсовой работы

Теоретический раздел

Раздел должен иметь наименование, отражающее теоретические аспекты темы курсовой работы. В общем случае в этом разделе должны быть отражены:

1) характеристика предмета исследования в соответствии темой курсовой работы, в частности, должен присутствовать анализ понятийной базы по заявленной теме.

2) анализ состояния вопроса, являющегося темой работы, при необходимости – методы анализа.

3) основные подходы к решению проблемы, являющейся темой курсовой работы.

Теоретический раздел должен давать ответы на вопросы, что представляет собой проблема, являющаяся темой курсовой работы.

Объем раздела – 10 - 12 с. В разделе должно быть **не менее двух и не более трех подразделов**.

Аналитический раздел

Наименование данного раздела должно быть связано с анализом состояния предмета исследования, являющегося темой курсовой работы.

Исследование проблемы должно быть целевым и глубоким.

В аналитическом разделе используются различные приемы анализа, указываются источники информации, приводятся аналитические таблицы и расчеты, графические способы отражения результатов анализа, делаются выводы.

В данном разделе обязательно использование не только статистического материала, но и результатов анализа, проведенного специалистами по данному вопросу, которые изложены в источниках периодической печати.

Объем аналитического раздела курсовой работы не должен превышать 10-12 страниц.

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

Перечень основной литературы:

1. Галатенко В.А. Основы информационной безопасности. – М.: Интернет-Университет Информационных Технологий, 2020.
2. Петраков А.В. Защита информации в компьютерных системах и сетях. – М.: Радио и связь, 2021.
3. Шаньгин В.Ф. Комплексная защита информации в корпоративных системах. – М.: ДМК Пресс, 2019.
4. Лопатин В.Н. Информационная безопасность России. – СПб.: Юридический центр Пресс, 2018.
5. Белов Е.Б., Лось В.П. Основы информационной безопасности. – М.: Горячая линия – Телеком, 2020.

Перечень дополнительной литературы:

6. Федеральный закон от 27.07.2006 № 149-ФЗ "Об информации, информационных технологиях и о защите информации".
7. Федеральный закон от 27.07.2006 № 152-ФЗ "О персональных данных".
8. Федеральный закон от 26.12.1995 № 208-ФЗ "О безопасности".
9. Приказы ФСТЭК России
10. ГОСТ Р 50922-2006 "Защита информации. Основные термины и определения".
11. ГОСТ Р 57580.1-2017 "Безопасность финансовых организаций".