Автономная некоммерческая организация высшего образования
«МОСКОВСКИЙ МЕЖДУНАРОДНЫЙ УНИВЕРСИТЕТ»

Рабочая программа дисциплины

Тестирование программного обеспечения

Направление подготовки	Информатика и вычислительная техника
Код	09.03.01
Направленность (профиль)	Автоматизированные системы обработки информации и управления
Квалификация выпускника	бакалавр

Москва
2026

1. Перечень кодов компетенций, формируемых дисциплиной в процессе освоения образовательной программы

Группа компетенций	Категория компетенций	Код
Профессиональные	-	ПК-2
Профессиональные	-	ПК-3

2. Компетенции и индикаторы их достижения

Код компетенции	Формулировка компетенции	Индикаторы достижения компетенции
ПК-2	**ПК-2.** Способен анализировать информационное обеспечение систем среднего и крупного масштаба сложности, реализовать инновационные решения по их интеграции, комплексному развитию и функциональной направленности	**ПК-2.1.** Знает методы подбора, обработки и анализа научно-технической и патентной информации по тематике исследования с использованием специализированных баз данных и информационных технологий **ПК-2.2.** Умеет осуществлять выбор эффективных средств и способов обеспечения качества разработки программного обеспечения, его нагрузочное, конфигурационное и интеграционное тестирование **ПК-2.3.** Владеет навыками создания программных продуктов, сетевых решений, в результате экспериментального исследования информационных систем, их математического описания, цифровых технологий
ПК-3	**ПК-3.** Способен управлять проектами в области информационных технологий на основе полученных планов проектов, поддерживать процессы разработки и модернизации	**ПК-3.1.** Знает первоначальные требования к ИС, перечень и последовательность работ, план мероприятий по управлению разработкой программного обеспечения, определение ресурсов, объемов работ по продвижению IT-продуктов **ПК-3.2.** Умеет проводить разработку архитектуры и прототипов информационных систем, их проектирование и дизайн, обеспечение кодирования на языках программирования, создавать пользовательскую документацию **ПК-3.3.** Владеет навыками проведения валидации программных продуктов и информационных систем, их интеграция с использованием инновационных аналитических методик

3. Описание планируемых результатов обучения по дисциплине

3.1. Описание планируемых результатов обучения по дисциплине

Планируемые результаты обучения по дисциплине представлены дескрипторами (знания, умения, навыки).

Дескрипторы по дисциплине	**Знать**	**Уметь**	**Владеть**
Код компетенции	**ПК-2**		
	Основы операционных систем: Знать архитектуру и основные компоненты операционных систем (например, Windows, Linux). Знать принципы работы файловых систем и управления процессами. Сетевые протоколы: Знать основные сетевые протоколы (TCP/IP, HTTP, HTTPS, FTP) и их функции. Знать принципы работы сетевых устройств (маршрутизаторов, коммутаторов, брандмауэров). Инструменты конфигурирования: Знать инструменты и утилиты для конфигурирования операционных систем и сетевых устройств (например, командная строка, PowerShell, SSH). Безопасность: Знать основные принципы и методы обеспечения безопасности операционных систем и сетевых устройств.	Конфигурировать операционные системы: Уметь устанавливать и настраивать операционные системы для веб-разработки. Уметь управлять пользователями и правами доступа в операционных системах. Настраивать сетевые устройства: Уметь конфигурировать маршрутизаторы и коммутаторы для обеспечения сетевой связности. Уметь настраивать брандмауэры для защиты веб-приложений. Использовать инструменты командной строки: Уметь использовать командную строку для выполнения задач администрирования и конфигурирования. Уметь применять скрипты для автоматизации процессов конфигурирования. Обеспечивать безопасность систем: Уметь применять методы шифрования и аутентификации для защиты данных. Уметь проводить аудит безопасности и выявлять уязвимости в системах.	Практическими навыками конфигурирования: Владеть навыками установки и настройки операционных систем в средах веб-разработки. Владеть навыками конфигурирования сетевых устройств для оптимизации работы веб-приложений. Работой с документацией: Владеть умением читать и интерпретировать техническую документацию по операционным системам и сетевым устройствам. Решением проблем: Владеть навыками диагностики и устранения неполадок в операционных системах и сетевых устройствах. Работой в команде: Владеть навыками взаимодействия с

			другими специалистами (разработчиками, системными администраторами) для эффективного конфигурирования и поддержки веб-приложений.
Код компетенции	**ПК-3**		
	- Знать основы математического аппарата Понимать ключевые математические концепции и методы, используемые для решения задач в области обработки информации, такие как алгебра, статистика и дискретная математика. - Знать методологию программирования Осознавать основные принципы и парадигмы программирования, включая объектно-ориентированное, функциональное и процедурное программирование. - Знать современные компьютерные технологии Иметь представление о современных технологиях и инструментах для работы с данными, таких как базы данных, облачные решения и языки программирования.	- Уметь применять математические методы Способность использовать математические модели и алгоритмы для решения практических задач, связанных с получением и обработкой информации. - Уметь разрабатывать программные решения Умение создавать программное обеспечение для автоматизации процессов получения, хранения и обработки данных с использованием различных языков программирования. - Уметь работать с компьютерными технологиями Способность эффективно использовать современные инструменты и технологии для управления данными, включая системы управления базами данных (СУБД) и средства визуализации данных.	- Владеть навыками программирования Умение писать код на нескольких языках программирования (например, Python, Java, C++) для решения задач обработки информации. - Владеть инструментами анализа данных Освоение специализированных программных средств для анализа, обработки и визуализации данных, таких как Excel, R или Tableau. - Владеть навыками проектирования систем Умение разрабатывать архитектуру информационных систем, учитывая требования к получению, хранению и передаче

			информации.

4. Место дисциплины (модуля) в структуре образовательной программы

Дисциплина «Тестирование программного обеспечения» относится к части, формируемой участниками образовательных отношений учебного плана ОПОП. Данная дисциплина взаимосвязана с другими дисциплинами, такими как: «Алгоритмизация и программирование», «Автоматизация офисных приложений», «Операционные системы и среды», «Математическая логика и дискретная математика», «Управление проектами».
В рамках освоения программы бакалавриата выпускники готовятся к решению задач профессиональной деятельности следующих типов:

Профиль (направленность) программы установлена путем ее ориентации на сферу профессиональной деятельности выпускников: Автоматизированные системы обработки информации и управления

5. Объем дисциплины

Виды учебной работы	***Формы обучения***
	Очно-заочное
Общая трудоемкость: зачетные единицы/часы	5/180
Контактная работа:	
Занятия лекционного типа	12
Занятия семинарского типа	16
Промежуточная аттестация: **зачет**	0,1
Самостоятельная работа (СРС)	151,9

6. Содержание дисциплины (модуля), структурированное по темам / разделам с указанием отведенного на них количества академических часов и видов учебных занятий

6.1. Распределение часов по разделам/темам и видам работы

6.1.1. Очно-заочная форма обучения

№ п/п	Раздел/тема	Виды учебной работы (в часах)						
		Контактная работа						Самостоятельная работа
		Занятия лекционного типа		Занятия семинарского типа				
		Лекции	*Иные учебные занятия*	*Практические занятия*	*Семинары*	*Лабораторные работы*	*Иные*	
1.	Основы тестирования ПО: цели, задачи, место в жизненном цикле разработки	1			2			25
2.	Классификация	1			2			25

	видов тестирования и выбор стратегии							
3.	Проектирование тест-кейсов: техники и шаблоны	2			2			25
4.	Юнит-тестирование и тестирование на уровне модулей	2			2			25
5.	Интеграционное и системное тестирование, работа с дефектами	2			3			25
6.	Нагрузочное и нефункциональное тестирование, оценка качества ПО	2			3			26,9
	Промежуточная аттестация	**0,1**						
	Итого	**12**			**16**			**151,9**

6.2 Программа дисциплины, структурированная по темам / разделам

6.2.1 Содержание лекционного курса

№ п/п	Наименование раздела дисциплины	Содержание раздела дисциплины
1.	Основы тестирования ПО: цели, задачи, место в жизненном цикле разработки	– Определение тестирования, отличие от отладки и верификации. – Цели тестирования: обнаружение дефектов, подтверждение соответствия требованиям, оценка рисков. – Место тестирования в моделях жизненного цикла (каскадная, итеративная, Agile, DevOps). – Уровни тестирования (модульное, интеграционное, системное, приёмочное) и их назначение. – Связь с дисциплинами «Проектирование ПО» и «Управление проектами»: трассировка требований, планирование тестов в спринте. – Базовые метрики: количество дефектов, плотность дефектов, покрытие требований.
2.	Классификация видов тестирования и выбор стратегии	– Функциональное и нефункциональное тестирование: примеры и области применения. – Виды по степени автоматизации: ручное, автоматизированное, полуавтоматизированное.

		– Тестирование по доступу к коду: «чёрный ящик», «белый ящик», «серый ящик». – Стратегии выбора тестов: риск-ориентированный подход, приоритизация на основе критичности функционала. – Практический аспект: как формировать стратегию для учебного проекта (ограниченные сроки, ограниченный объём кода).
3.	Проектирование тест-кейсов: техники и шаблоны	– Структура тест-кейса и чек-листа: обязательные и опциональные поля. – Техники проектирования: классы эквивалентности, анализ граничных значений, таблицы решений, диаграммы причин-следствий. – Примеры применения техник к типовым задачам (формы ввода, валидация данных, бизнес-правила). – Шаблоны документации: матрица трассируемости требований к тест-кейсам. – Критерии качества тест-кейса: однозначность, воспроизводимость, независимость.
4.	Юнит-тестирование и тестирование на уровне модулей	– Понятие юнит-теста, изоляция тестируемого кода, моки и стабы. – Фреймворки для юнит-тестирования (JUnit, pytest, NUnit) и базовые паттерны написания тестов. – Критерии покрытия: statement coverage, branch coverage, path coverage; связь с цикломатической сложностью из курса «Алгоритмы и структуры данных». – Практика: написание тестов для простых функций (валидация, вычисления, обработка исключений). – Интеграция с CI/CD: запуск тестов при сборке, интерпретация отчётов.
5.	Интеграционное и системное тестирование, работа с дефектами	– Цели и задачи интеграционного тестирования: проверка взаимодействия компонентов, контрактов API, очередей сообщений. – Подходы к интеграции: «снизу вверх», «сверху вниз», «большой взрыв». – Системное тестирование: сценарии использования, проверка сквозного функционала. – Жизненный цикл дефекта: обнаружение, оформление, приоритизация, исправление, ретест, закрытие. – Инструменты баг-трекинга, поля дефекта, критерии «готов к тестированию» и «готов к релизу».

6.	Нагрузочное и нефункциональное тестирование, оценка качества ПО	– Виды нефункционального тестирования: производительность, надёжность, безопасность, удобство использования. – Нагрузочное тестирование: типы нагрузок (пик, стресс, долговременная), метрики (RPS, latency, error rate). – Инструменты: Apache JMeter, k6, Locust; базовые сценарии и интерпретация результатов. – Стандарты качества ПО (ISO/IEC 25000), характеристики качества и измеримые метрики. – Итоговая стратегия тестирования для учебного проекта: баланс между функционалом, рисками и ресурсами.

6.2.2 Содержание практических занятий

№ п/п	Наименование раздела дисциплины	Содержание практических занятий
1.	Основы тестирования ПО: цели, задачи, место в жизненном цикле разработки	– Регистрация и работа в баг-трекинговой системе (Jira/YouTrack/Redmine — в зависимости от принятого в вузе стека). – Заполнение шаблона тест-кейса: идентификатор, предусловие, шаги, ожидаемый результат, статус. – Оформление дефекта по шаблону: шаги воспроизведения, окружение, ожидаемое/фактическое поведение. – Практика: на примере простого веб-интерфейса (или консольного приложения) оформить 3–5 тест-кейсов и 1 дефект. – Обсуждение типичных ошибок: слишком общие шаги, отсутствие предусловий, неоднозначные ожидаемые результаты.
2.	Классификация видов тестирования и выбор стратегии	– Отработка техник: классы эквивалентности и граничные значения на примерах валидации полей (возраст, диапазон цен, формат даты). – Построение таблицы решений для бизнес-правила (например, расчёт скидки по условиям). – Мини-задача: для заданной функции (на псевдокоде или в коде) спроектировать набор тестов и обосновать покрытие. – Расчёт минимального набора тестов для покрытия всех классов эквивалентности и граничных случаев.
3.	Проектирование тест-кейсов: техники и шаблоны	– На основе небольшого набора требований (5–7 пунктов) построить матрицу трассируемости: требование → тест-кейс → статус.

		– Для каждого требования выбрать технику проектирования и обосновать выбор. – Рассчитать покрытие требований (в процентах) и выявить «непокрытые» требования. – Переписать неоднозначные тест-кейсы, чтобы они соответствовали критериям качества (однозначность, воспроизводимость).
4.	Юнит-тестирование и тестирование на уровне модулей	– Выбор фреймворка под язык курса программирования (Java/Python/C#). – Написание юнит-тестов для 3–4 функций: валидация, арифметические операции, обработка исключений. – Использование моков для зависимостей (пример с подменой внешнего сервиса или базы данных). – Анализ отчёта о покрытии (JaCoCo, coverage.py) и доработка тестов для повышения покрытия. – Добавление тестов в репозиторий (Git) и проверка запуска через простой CI-скрипт (GitHub Actions/GitLab CI — упрощённый пример). – Связь с курсом «Программирование»: тесты как часть репозитория, коммиты и ветки.
5.	Юнит-тестирование и тестирование на уровне модулей	– Настройка простого пайплайна CI/CD (GitHub Actions / GitLab CI): триггер на пуш, установка зависимостей, запуск тестов. – Написание набора регрессионных тест-кейсов для ранее протестированных функций: проверка, что старые функции не «сломались» при добавлении новой функциональности. – Разделение тестов по уровням приоритета (высокий/средний/низкий) для оптимизации времени прогона в CI. – Работа с артефактами: сохранение отчётов о покрытии и логов тестов, анализ причин падений в CI. – Практика: добавить в учебный репозиторий YAML-файл пайплайна и убедиться, что тесты запускаются автоматически; разобрать 1–2 сценария падения сборки.
6.	Интеграционное и системное тестирование, работа с дефектами	– Проектирование интеграционных тестов для пары компонентов (сервис + БД, микросервис + очередь). – Работа с API: отправка запросов (GET/POST), проверка статусов и JSON-ответов. – Проверка контрактов: соответствие схемы, валидность данных, обработка ошибок. – Чтение и анализ логов приложения и сервера: сопоставление логов с шагами тест-кейса для локализации проблемы.

		– Формулирование гипотез о причинах дефекта и план ретеста после исправления. – Инструменты: Postman, curl, pytest + requests; разбор типовых сценариев.
7.	Интеграционное и системное тестирование, работа с дефектами	– Введение в базовые уязвимости: инъекции, переполнение буфера, некорректная валидация ввода; обсуждение, как тестирование помогает выявлять такие проблемы без проведения полноценного пентеста. – Техники негативного тестирования: подбор некорректных, граничных и экстремальных входных данных (пустые значения, спецсимволы, очень большие числа, неверные типы). – Проектирование тест-кейсов «негативного» сценария для API и форм ввода: проверка обработки ошибок, сообщений пользователю, целостности данных. – Проверка обработки исключений и отказоустойчивости: что происходит при отказе внешней зависимости (БД, внешнего API). – Инструменты: Postman (скрипты проверок), простые статические проверки (lint), анализ сообщений об ошибках. – Практика: для 2–3 эндпоинтов API или полей формы подготовить набор негативных тест-кейсов, запустить их и оформить найденные дефекты (шаги, ожидаемое/фактическое, приоритет).
8.	Нагрузочное и нефункциональное тестирование, оценка качества ПО	– Установка и настройка инструмента (Apache JMeter или k6). – Создание сценария: запросы к эндпоинту, параметры нагрузки (количество пользователей, длительность). – Запуск теста, сбор метрик (время ответа, ошибки, пропускная способность). – Интерпретация графиков и отчётов: выявление «узких мест» и формулирование рекомендаций. – Подготовка итогового отчёта по тестированию: план, набор тестов, результаты, дефекты, метрики качества. – Защита мини-проекта: презентация стратегии и результатов тестирования, ответы на вопросы о выборе приоритетов и компромиссах.

6.2.3 Содержание самостоятельной работы

№ п/п	**Наименование раздела дисциплины**	**Содержание раздела дисциплины**
1.	Основы тестирования ПО: цели, задачи, место в	– Определение тестирования, отличие от отладки и верификации.

	жизненном цикле разработки	– Цели тестирования: обнаружение дефектов, подтверждение соответствия требованиям, оценка рисков. – Место тестирования в моделях жизненного цикла (каскадная, итеративная, Agile, DevOps). – Уровни тестирования (модульное, интеграционное, системное, приёмочное) и их назначение. – Связь с дисциплинами «Проектирование ПО» и «Управление проектами»: трассировка требований, планирование тестов в спринте. – Базовые метрики: количество дефектов, плотность дефектов, покрытие требований.
2.	Классификация видов тестирования и выбор стратегии	– Функциональное и нефункциональное тестирование: примеры и области применения. – Виды по степени автоматизации: ручное, автоматизированное, полуавтоматизированное. – Тестирование по доступу к коду: «чёрный ящик», «белый ящик», «серый ящик». – Стратегии выбора тестов: риск-ориентированный подход, приоритизация на основе критичности функционала. – Практический аспект: как формировать стратегию для учебного проекта (ограниченные сроки, ограниченный объём кода).
3.	Проектирование тест-кейсов: техники и шаблоны	– Структура тест-кейса и чек-листа: обязательные и опциональные поля. – Техники проектирования: классы эквивалентности, анализ граничных значений, таблицы решений, диаграммы причин-следствий. – Примеры применения техник к типовым задачам (формы ввода, валидация данных, бизнес-правила). – Шаблоны документации: матрица трассируемости требований к тест-кейсам. – Критерии качества тест-кейса: однозначность, воспроизводимость, независимость.
4.	Юнит-тестирование и тестирование на уровне модулей	– Понятие юнит-теста, изоляция тестируемого кода, моки и стабы. – Фреймворки для юнит-тестирования (JUnit, pytest, NUnit) и базовые паттерны написания тестов. – Критерии покрытия: statement coverage, branch coverage, path coverage; связь с цикломатической сложностью из курса «Алгоритмы и структуры данных».

		– Практика: написание тестов для простых функций (валидация, вычисления, обработка исключений). – Интеграция с CI/CD: запуск тестов при сборке, интерпретация отчётов.
5.	Интеграционное и системное тестирование, работа с дефектами	– Цели и задачи интеграционного тестирования: проверка взаимодействия компонентов, контрактов API, очередей сообщений. – Подходы к интеграции: «снизу вверх», «сверху вниз», «большой взрыв». – Системное тестирование: сценарии использования, проверка сквозного функционала. – Жизненный цикл дефекта: обнаружение, оформление, приоритизация, исправление, ретест, закрытие. – Инструменты баг-трекинга, поля дефекта, критерии «готов к тестированию» и «готов к релизу».
6.	Нагрузочное и нефункциональное тестирование, оценка качества ПО	– Виды нефункционального тестирования: производительность, надёжность, безопасность, удобство использования. – Нагрузочное тестирование: типы нагрузок (пик, стресс, долговременная), метрики (RPS, latency, error rate). – Инструменты: Apache JMeter, k6, Locust; базовые сценарии и интерпретация результатов. – Стандарты качества ПО (ISO/IEC 25000), характеристики качества и измеримые метрики. – Итоговая стратегия тестирования для учебного проекта: баланс между функционалом, рисками и ресурсами.

7. Текущий контроль по дисциплине (модулю) в рамках учебных занятий

В рамках текущего контроля преподаватель самостоятельно может проводить следующие мероприятия:

№ п/п	Контролируемые разделы (темы)	Наименование оценочного средства
1.	Основы тестирования ПО: цели, задачи, место в жизненном цикле разработки	Опрос, информационный проект.
2.	Классификация видов тестирования и выбор стратегии	Опрос, информационный проект, тестирование.
3.	Проектирование тест-кейсов: техники и шаблоны	Опрос, информационный проект.

4.	Юнит-тестирование и тестирование на уровне модулей	Опрос, информационный проект.
5.	Интеграционное и системное тестирование, работа с дефектами	Опрос, информационный проект, тестирование.
6.	Нагрузочное и нефункциональное тестирование, оценка качества ПО	Опрос, информационный проект.

8. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины (модуля)

8.1. Основная литература:

1. Аниче, М. Эффективное тестирование программного обеспечения : практическое пособие / М. Аниче ; пер. с англ. А. Н. Киселёва. — Москва : ДМК Пресс, 2023. — 1 URL: https://www.iprbookshop.ru/159939.html
2. Бубнов, А. А. Тестирование программного обеспечения : учебник / А. А. Бубнов, К. А. Реутский, В. В. Тишкина. — 2024. — 1 URL: https://www.iprbookshop.ru/144824.html
3. Котляров, В. П. Основы тестирования программного обеспечения : учебное пособие для СПО / В. П. Котляров. — 2-е изд. — Саратов : Профобразование, 2025. — 336 с. — ISBN 978-5-4488-0364-2. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/153351.html
4. Николюкин, М. С. Тестирование программного обеспечения : учебное пособие / М. С. Николюкин, В. В. Конкина, К. И. Патутин. — Тамбов : Тамбовский государственный технический университет, ЭБС АСВ, 2025. — 1 URL: https://www.iprbookshop.ru/154963.html
5. Проскуряков, А. В. Качество и тестирование программного обеспечения. Метрология программного обеспечения : учебное пособие / А. В. Проскуряков. — Ростов-на-Дону, Таганрог : Издательство Южного федерального университета, 2022. — 197 с. — ISBN 978-5-9275-4044-0. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/125702.html

1.

2. 8.2. Дополнительная литература:

1. Базовые принципы разработки программного обеспечения : учебное пособие / В. И. Шипков, Т. Р. Захаренкова, А. А. Нечаев, А. С. Грицай. — Омск : Омский государственный технический университет, 2023. — 116 с. — ISBN 978-5-8149-3671-4. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/140826.html
2. Бубнов, А. А. Тестирование программного обеспечения : учебное пособие / А. А. Бубнов, В. В. Тишкина. — Рязань : Рязанский государственный радиотехнический университет, 2024. — 1 URL: https://www.iprbookshop.ru/150311.html
3. Синицын, С. В. Основы разработки программного обеспечения на примере языка С : учебник / С. В. Синицын, О. И. Хлытчиев. — 4-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2025. — 211 с. — ISBN 978-5-4497-0916-5. — Текст : электронный // Цифровой

образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/146374.html
4. Тишина, Н. А. Современные средства разработки программного обеспечения : учебное пособие / Н. А. Тишина, Е. Н. Чернопрудова, В. Н. Костин. — Оренбург : Оренбургский государственный университет, ЭБС АСВ, 2024. — 191 с. — ISBN 978-5-7410-3274-9. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/153085.html (дата обращения: 30.06.2026). — Режим доступа: для авторизир. пользователей

10. Методические указания для обучающихся по освоению дисциплины (модуля)

Успешное освоение данного курса базируется на рациональном сочетании нескольких видов учебной деятельности – лекций, семинарских занятий, самостоятельной работы. При этом самостоятельную работу следует рассматривать одним из главных звеньев полноценного высшего образования, на которую отводится значительная часть учебного времени.

Самостоятельная работа студентов складывается из следующих составляющих:

1. Работа с основной и дополнительной литературой, с материалами интернета и конспектами лекций;
2. Внеаудиторная подготовка к контрольным работам, выполнение докладов, рефератов и курсовых работ;
3. Выполнение самостоятельных практических работ;
4. Подготовка к экзаменам (зачетам) непосредственно перед ними.

Для правильной организации работы необходимо учитывать порядок изучения разделов курса, находящихся в строгой логической последовательности. Поэтому хорошее усвоение одной части дисциплины является предпосылкой для успешного перехода к следующей. Задания, проблемные вопросы, предложенные для изучения дисциплины, в том числе и для самостоятельного выполнения, носят междисциплинарный характер и базируются, прежде всего, на причинно-следственных связях между компонентами окружающего нас мира. Важным составляющим в изучении данного курса является решение ситуационных задач и работа над проблемно-аналитическими заданиями, что предполагает знание соответствующей научной терминологии и т.д.

Для лучшего запоминания материала целесообразно использовать индивидуальные особенности и разные виды памяти: зрительную, слуховую, ассоциативную. Успешному запоминанию также способствует приведение ярких свидетельств и наглядных примеров. Учебный материал должен постоянно повторяться и закрепляться.

При выполнении докладов, творческих, информационных, исследовательских проектов особое внимание следует обращать на подбор источников информации и методику работы с ними.

Для успешной сдачи экзамена (зачета) рекомендуется соблюдать следующие правила:

1. Подготовка к экзамену (зачету) должна проводиться систематически, в течение всего семестра.
2. Интенсивная подготовка должна начаться не позднее, чем за месяц до экзамена.
3. Время непосредственно перед экзаменом (зачетом) лучше использовать таким образом, чтобы оставить последний день свободным для повторения курса в целом, для систематизации материала и доработки отдельных вопросов.

На экзамене высокую оценку получают студенты, использующие данные, полученные в процессе выполнения самостоятельных работ, а также использующие собственные выводы на основе изученного материала.

Учитывая значительный объем теоретического материала, студентам рекомендуется регулярное посещение и подробное конспектирование лекций.

11. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень программного обеспечения и информационных справочных систем (при необходимости)

1. Microsoft Windows Server;

2.Семейство ОС Microsoft Windows;

3. Libre Office свободно распространяемый офисный пакет с открытым исходным кодом;

4. Информационно-справочная система: Система КонсультантПлюс (КонсультантПлюс);

5. Информационно-правовое обеспечение Гарант: Электронный периодический справочник «Система ГАРАНТ» (Система ГАРАНТ);

Перечень используемого программного обеспечения указан в п.12 данной рабочей программы дисциплины.

12. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

12.1. Учебная аудитория для проведения учебных занятий, предусмотренных образовательной программой, оснащенная оборудованием и техническими средствами обучения.

Специализированная мебель:

Комплект учебной мебели (стол, стул) по количеству обучающихся; комплект мебели для преподавателя; доска (маркерная).

Технические средства обучения:

Компьютер в сборе для преподавателя, проектор, экран, колонки

Перечень лицензионного программного обеспечения, в том числе отечественного производства:

Windows 10, КонсультантПлюс, Kaspersky Endpoint Security.

Перечень свободно распространяемого программного обеспечения:

Яндекс Браузер, LibreOffice, МТС Линк.

Подключение к сети «Интернет» и обеспечение доступа в электронную информационно-образовательную среду ММУ.

12.2. Помещение для самостоятельной работы обучающихся.

Специализированная мебель:

Комплект учебной мебели (стол, стул) по количеству обучающихся; комплект мебели для преподавателя; доска (маркерная).

Технические средства обучения:

Компьютер в сборе для преподавателя; компьютеры в сборе для обучающихся; колонки; проектор, экран.

Перечень лицензионного программного обеспечения, в том числе отечественного производства:

Windows Server 2016, Windows 10, Microsoft Office, КонсультантПлюс, Kaspersky Endpoint Security.

Перечень свободно распространяемого программного обеспечения:

Яндекс Браузер, LibreOffice, МТС Линк, Gimp, Paint.net, AnyLogic, Inkscape.

Помещение для самостоятельной работы обучающихся оснащено компьютерной техникой с возможностью подключения к сети "Интернет" и обеспечением доступа в электронную информационно-образовательную среду ММУ.

13.Образовательные технологии, используемые при освоении дисциплины

Для освоения дисциплины используются как традиционные формы занятий – лекции (типы лекций – установочная, вводная, текущая, заключительная, обзорная; виды лекций – проблемная, визуальная, лекция конференция, лекция консультация); и семинарские (практические) занятия, так и активные и интерактивные формы занятий - деловые и ролевые игры, решение ситуационных задач и разбор конкретных ситуаций.

На учебных занятиях используются технические средства обучения мультимедийной аудитории: компьютер, монитор, колонки, настенный экран, проектор, микрофон, пакет программ Microsoft Office для демонстрации презентаций и медиафайлов, видеопроектор для демонстрации слайдов, видеосюжетов и др. Тестирование обучаемых может осуществляться с использованием компьютерного оборудования университета.

13.1. В освоении учебной дисциплины используются следующие традиционные образовательные технологии:

- чтение проблемно-информационных лекций с использованием доски и видеоматериалов;
- семинарские занятия для обсуждения, дискуссий и обмена мнениями;
- контрольные опросы;
- консультации;
- самостоятельная работа студентов с учебной литературой и первоисточниками;
- подготовка и обсуждение рефератов (проектов), презентаций (научно-исследовательская работа);
- тестирование по основным темам дисциплины.

13.2. Активные и интерактивные методы и формы обучения

Из перечня видов: («*мозговой штурм», анализ НПА, анализ проблемных ситуаций, анализ конкретных ситуаций, инциденты, имитация коллективной профессиональной деятельности, разыгрывание ролей, творческая работа, связанная с освоением дисциплины, ролевая игра, круглый стол, диспут, беседа, дискуссия, мини-конференция и др.*) используются следующие:

- *диспут;*
- *анализ проблемных, творческих заданий, ситуационных задач;*
- *ролевая игра;*
- *круглый стол;*
- *мини-конференция;*
- *дискуссия;*
- *беседа.*

13.3. Особенности обучения инвалидов и лиц с ограниченными возможностями здоровья (ОВЗ)

При организации обучения по дисциплине учитываются особенности организации взаимодействия с инвалидами и лицами с ограниченными возможностями здоровья (далее – инвалиды и лица с ОВЗ) с целью обеспечения их прав. При обучении учитываются особенности их психофизического развития, индивидуальные возможности и при необходимости обеспечивается коррекция нарушений развития и социальная адаптация указанных лиц.

Выбор методов обучения определяется содержанием обучения, уровнем методического и материально-технического обеспечения, особенностями восприятия учебной информации студентов с инвалидностью и студентов с ограниченными возможностями здоровья и т.д. В образовательном процессе используются социально-активные и рефлексивные методы обучения, технологии социокультурной реабилитации

с целью оказания помощи в установлении полноценных межличностных отношений с другими студентами, создании комфортного психологического климата в студенческой группе.

При обучении лиц с ограниченными возможностями здоровья электронное обучение и дистанционные образовательные технологии предусматривают возможность приема-передачи информации в доступных для них формах.

Обучающиеся из числа лиц с ограниченными возможностями здоровья обеспечены печатными и электронными образовательными ресурсами в формах, адаптированных к ограничениям их здоровья.

Приложение

Автономная некоммерческая организация высшего образования
«МОСКОВСКИЙ МЕЖДУНАРОДНЫЙ УНИВЕРСИТЕТ»

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ДИСЦИПЛИНЕ

Тестирование программного обеспечения

Направление подготовки	Информатика и вычислительная техника
Код	09.03.01
Направленность (профиль)	Автоматизированные системы обработки информации и управления
Квалификация выпускника	бакалавр

Москва
2026

1. **Перечень кодов компетенций, формируемых дисциплиной в процессе освоения образовательной программы**

Группа компетенций	Категория компетенций	Код
Профессиональные	-	ПК-2
Профессиональные	-	ПК-3

2. **Компетенции и индикаторы их достижения**

Код компетенции	Формулировка компетенции	Индикаторы достижения компетенции
ПК-2	**ПК-2.** Способен анализировать информационное обеспечение систем среднего и крупного масштаба сложности, реализовать инновационные решения по их интеграции, комплексному развитию и функциональной направленности	**ПК-2.1.** Знает методы подбора, обработки и анализа научно-технической и патентной информации по тематике исследования с использованием специализированных баз данных и информационных технологий **ПК-2.2.** Умеет осуществлять выбор эффективных средств и способов обеспечения качества разработки программного обеспечения, его нагрузочное, конфигурационное и интеграционное тестирование **ПК-2.3.** Владеет навыками создания программных продуктов, сетевых решений, в результате экспериментального исследования информационных систем, их математического описания, цифровых технологий
ПК-3	**ПК-3.** Способен управлять проектами в области информационных технологий на основе полученных планов проектов, поддерживать процессы разработки и модернизации	**ПК-3.1.** Знает первоначальные требования к ИС, перечень и последовательность работ, план мероприятий по управлению разработкой программного обеспечения, определение ресурсов, объемов работ по продвижению IT-продуктов **ПК-3.2.** Умеет проводить разработку архитектуры и прототипов информационных систем, их проектирование и дизайн, обеспечение кодирования на языках программирования, создавать пользовательскую документацию **ПК-3.3.** Владеет навыками проведения валидации программных продуктов и информационных систем, их интеграция с использованием инновационных аналитических методик

3. **Описание планируемых результатов обучения по дисциплине**

3.1. Описание планируемых результатов обучения по дисциплине

Планируемые результаты обучения по дисциплине представлены дескрипторами (знания, умения, навыки).

Дескрипторы по дисциплине	**Знать**	**Уметь**	**Владеть**
Код компетенции	**ПК-2**		
	Основы операционных систем: Знать архитектуру и основные компоненты операционных систем (например, Windows, Linux). Знать принципы работы файловых систем и управления процессами. Сетевые протоколы: Знать основные сетевые протоколы (TCP/IP, HTTP, HTTPS, FTP) и их функции. Знать принципы работы сетевых устройств (маршрутизаторов, коммутаторов, брандмауэров). Инструменты конфигурирования: Знать инструменты и утилиты для конфигурирования операционных систем и сетевых устройств (например, командная строка, PowerShell, SSH). Безопасность: Знать основные принципы и методы обеспечения безопасности операционных систем и сетевых устройств.	Конфигурировать операционные системы: Уметь устанавливать и настраивать операционные системы для веб-разработки. Уметь управлять пользователями и правами доступа в операционных системах. Настраивать сетевые устройства: Уметь конфигурировать маршрутизаторы и коммутаторы для обеспечения сетевой связности. Уметь настраивать брандмауэры для защиты веб-приложений. Использовать инструменты командной строки: Уметь использовать командную строку для выполнения задач администрирования и конфигурирования. Уметь применять скрипты для автоматизации процессов конфигурирования. Обеспечивать безопасность систем: Уметь применять методы шифрования и аутентификации для защиты данных. Уметь проводить аудит безопасности и выявлять уязвимости в системах.	Практическими навыками конфигурирования: Владеть навыками установки и настройки операционных систем в средах веб-разработки. Владеть навыками конфигурирования сетевых устройств для оптимизации работы веб-приложений. Работой с документацией: Владеть умением читать и интерпретировать техническую документацию по операционным системам и сетевым устройствам. Решением проблем: Владеть навыками диагностики и устранения неполадок в операционных системах и сетевых устройствах. Работой в команде: Владеть навыками взаимодействия с другими

			специалистами (разработчиками, системными администраторам и) для эффективного конфигурировани я и поддержки веб-приложений.
Код компетенции	**ПК-3**		
	- Знать основы математического аппарата Понимать ключевые математические концепции и методы, используемые для решения задач в области обработки информации, такие как алгебра, статистика и дискретная математика. - Знать методологию программирования Осознавать основные принципы и парадигмы программирования, включая объектно-ориентированное, функциональное и процедурное программирование. - Знать современные компьютерные технологии Иметь представление о современных технологиях и инструментах для работы с данными, таких как базы данных, облачные решения и языки программирования.	- Уметь применять математические методы Способность использовать математические модели и алгоритмы для решения практических задач, связанных с получением и обработкой информации. - Уметь разрабатывать программные решения Умение создавать программное обеспечение для автоматизации процессов получения, хранения и обработки данных с использованием различных языков программирования. - Уметь работать с компьютерными технологиями Способность эффективно использовать современные инструменты и технологии для управления данными, включая системы управления базами данных (СУБД) и средства визуализации данных.	- Владеть навыками программировани я Умение писать код на нескольких языках программировани я (например, Python, Java, C++) для решения задач обработки информации. - Владеть инструментами анализа данных Освоение специализированн ых программных средств для анализа, обработки и визуализации данных, таких как Excel, R или Tableau. - Владеть навыками проектирования систем Умение разрабатывать архитектуру информационных систем, учитывая требования к получению, хранению и передаче информации.

3.2. Критерии оценки знаний студентов

Шкала оценивания	Индикаторы достижения	Показатели оценивания результатов обучения
ОТЛИЧНО/Зачтено	Знает:	- студент глубоко и всесторонне усвоил материал, уверенно, логично, последовательно и грамотно его излагает, опираясь на знания основной и дополнительной литературы, - на основе системных научных знаний делает квалифицированные выводы и обобщения, свободно оперирует категориями и понятиями.
	Умеет:	- студент умеет самостоятельно и правильно решать учебно-профессиональные задачи или задания, уверенно, логично, последовательно и аргументировано излагать свое решение, используя научные понятия, ссылаясь на нормативную базу.
	Владеет:	- студент владеет рациональными методами (с использованием рациональных методик) решения сложных профессиональных задач, представленных деловыми играми, кейсами и т.д.; При решении продемонстрировал навыки - выделения главного, - связкой теоретических положений с требованиями руководящих документов, - изложения мыслей в логической последовательности, - самостоятельного анализа факты, событий, явлений, процессов в их взаимосвязи и диалектическом развитии.
ХОРОШО/Зачтено	Знает:	- студент твердо усвоил материал, достаточно грамотно его излагает, опираясь на знания основной и дополнительной литературы, - затрудняется в формулировании квалифицированных выводов и обобщений, оперирует категориями и понятиями, но не всегда правильно их верифицирует.
	Умеет:	- студент умеет самостоятельно и в основном правильно решать учебно-профессиональные задачи или задания, уверенно, логично, последовательно и аргументировано излагать свое решение, не в полной мере используя научные понятия и ссылки на нормативную базу.
	Владеет:	- студент в целом владеет рациональными методами решения сложных профессиональных задач, представленных деловыми играми, кейсами и т.д.; При решении смог продемонстрировать достаточность, но не глубинность навыков - выделения главного, - изложения мыслей в логической последовательности. - связки теоретических положений с требованиями руководящих документов, - самостоятельного анализа факты, событий, явлений, процессов в их взаимосвязи и диалектическом развитии.

УДОВЛЕТВОРИТЕЛЬНО/Зачтено	Знает:	- студент ориентируется в материале, однако затрудняется в его изложении; - показывает недостаточность знаний основной и дополнительной литературы; - слабо аргументирует научные положения; - практически не способен сформулировать выводы и обобщения; - частично владеет системой понятий.
	Умеет:	- студент в основном умеет решить учебно-профессиональную задачу или задание, но допускает ошибки, слабо аргументирует свое решение, недостаточно использует научные понятия и руководящие документы.
	Владеет:	- студент владеет некоторыми рациональными методами решения сложных профессиональных задач, представленных деловыми играми, кейсами и т.д.; При решении продемонстрировал недостаточность навыков - выделения главного, - изложения мыслей в логической последовательности. - связки теоретических положений с требованиями руководящих документов, - самостоятельного анализа факты, событий, явлений, процессов в их взаимосвязи и диалектическом развитии.
НЕУДОВЛЕТВОРИТЕЛЬНО/Не зачтено	Знает:	- студент не усвоил значительной части материала; - не может аргументировать научные положения; - не формулирует квалифицированных выводов и обобщений; - не владеет системой понятий.
	Умеет:	студент не показал умение решать учебно-профессиональную задачу или задание.
	Владеет:	не выполнены требования, предъявляемые к навыкам, оцениваемым “удовлетворительно”.

При ответе на вопросы в рамках прохождения промежуточной аттестации (зачет/ зачет с оценкой/ экзамен) допускается вольная формулировка ответа, по смыслу раскрывающая содержание ответа, указанного в фонде оценочных средств, в качестве верного ответа.

4. Типовые контрольные задания (закрытого, открытого и иного типа) для проведения промежуточной аттестации, необходимые для оценки достижения компетенции, соотнесенной с результатами обучения по дисциплине

ТЕСТИРОВАНИЕ

5 СЕМЕСТР
ПК-2

1. Что является основной целью тестирования ПО?
А) Исправить все ошибки в коде.
Б) Обнаружить дефекты и оценить риски, подтвердить соответствие требованиям.
В) Заменить процесс отладки.
Г) Подтвердить, что ПО идеально и не содержит ошибок.

2. Чем тестирование принципиально отличается от отладки?
А) Тестирование и отладка — это одно и то же.
Б) Тестирование выявляет дефекты, отладка устраняет их.
В) Отладка выявляет дефекты, тестирование устраняет их.
Г) Тестирование проводится только после релиза, отладка — до.

3. Какой уровень тестирования проверяет взаимодействие между модулями?
А) Модульное тестирование.
Б) Интеграционное тестирование.
В) Системное тестирование.
Г) Приёмочное тестирование.

4. В какой модели жизненного цикла тестирование наиболее тесно интегрировано на каждом этапе и предусмотрено непрерывное тестирование?
А) Каскадная модель.
Б) Итеративная модель.
В) DevOps/Agile.
Г) Прототипная модель.

5. Что означает термин «покрытие требований» в тестировании?
А) Доля требований, для которых подготовлены и выполнены тест-кейсы.
Б) Количество найденных дефектов.
В) Процент строк кода, выполненных при прогоне тестов.
Г) Количество тест-кейсов в наборе.

6. Какой вид тестирования относится к нефункциональному?
А) Проверка расчёта скидки по бизнес-правилу.
Б) Проверка корректности формы входа.
В) Проверка времени отклика при 1000 одновременных пользователей.
Г) Проверка валидации поля «Email».

7. Что подразумевает подход «тестирование белого ящика»?
А) Тестировщик не знает внутренней структуры кода.
Б) Тестировщик опирается на структуру кода (ветвления, циклы, пути).
В) Тестируются только публичные API.
Г) Тесты пишутся исключительно по документации.

8. Какой подход к выбору тестов наиболее уместен при ограниченных сроках и высоком риске для бизнеса?
А) Протестировать всё подряд.
Б) Риск-ориентированный подход: приоритет на критические сценарии.

В) Случайная выборка тест-кейсов.
Г) Протестировать только новые функции.

9. Что такое «негативное тестирование»?
А) Проверка поведения системы на некорректных, граничных и экстремальных данных.
Б) Тестирование только негативных сценариев без позитивных.
В) Тестирование с целью доказать, что система плохая.
Г) Тестирование, при котором не фиксируется результат.

10. Какой тип тестирования лучше всего подходит для проверки контрактов API?
А) Юнит-тестирование.
Б) Интеграционное тестирование.
В) Нагрузочное тестирование.
Г) Статическое тестирование.

11. Для чего применяется техника «классы эквивалентности»?
А) Чтобы сгруппировать входные данные в наборы, где ожидается одинаковое поведение системы.
Б) Чтобы покрыть все возможные комбинации входных параметров.
В) Чтобы протестировать только граничные значения.
Г) Чтобы оценить сложность алгоритма.

12. Какой из следующих элементов НЕ является обязательным в тест-кейсе?
А) Предусловие.
Б) Шаги теста.
В) Ожидаемый результат.
Г) Фактический результат. (Он заполняется при выполнении теста.)

13. Что позволяет выявить техника «анализ граничных значений»?
А) Ошибки на границах диапазонов (например, минимальное и максимальное допустимое значение).
Б) Ошибки в середине диапазона.
В) Ошибки производительности.
Г) Ошибки архитектуры.

14. Что такое матрица трассируемости требований?
А) Таблица, связывающая требования с тест-кейсами для оценки покрытия.
Б) Список всех найденных дефектов.
В) План релизов.
Г) Диаграмма классов системы.

15. Какой инструмент лучше всего использовать для визуализации бизнес-правил и проектирования тестов на их основе?
А) JMeter.
Б) Jira.
В) Таблица решений.
Г) Git.

ПК-3

1. Что такое мок (mock) в юнит-тестировании?
А) Объект-заглушка, имитирующий поведение зависимости для изоляции

тестируемого кода.
Б) Реальная база данных.
В) Полный экземпляр внешней службы.
Г) Логирование всех вызовов.

2. Какой критерий покрытия оценивает, были ли выполнены все ветви (условия) в коде?
А) Statement coverage.
Б) Branch coverage.
В) Path coverage.
Г) Function coverage.

3. Почему важно изолировать юнит-тест от внешних зависимостей?
А) Чтобы тест был быстрым, детерминированным и не зависел от состояния БД/сети.
Б) Потому что внешние зависимости всегда работают корректно.
В) Чтобы увеличить количество дефектов.
Г) Это не обязательно.

4. Какой фреймворк используется для написания юнит-тестов в Python?
А) JUnit.
Б) pytest.
В) NUnit.
Г) Selenium.

5. Что показывает цикломатическая сложность?
А) Количество независимых путей в графе потока управления, что помогает оценить необходимый минимум тестов.
Б) Время выполнения функции.
В) Количество строк кода.
Г) Количество внешних зависимостей.

6. Какой подход предполагает поэтапное объединение модулей и их тестирование по мере интеграции?
А) «Снизу вверх».
Б) «Большой взрыв».
В) Только системное тестирование.
Г) Только юнит-тесты.

7. Что включает в себя жизненный цикл дефекта?
А) Только обнаружение и исправление.
Б) Обнаружение, оформление, приоритизация, исправление, ретест, закрытие.
В) Только оформление и закрытие.
Г) Обнаружение и немедленное закрытие.

8. Что означает статус дефекта «Ready for Testing»?
А) Разработчик исправил дефект, и он готов к проверке тестировщиком.
Б) Дефект только что обнаружен.
В) Дефект закрыт.
Г) Дефект не будет исправлен.

9. Для чего используются логи при локализации дефекта?
А) Чтобы сопоставить шаги теста с событиями в системе и найти причину ошибки.
Б) Чтобы скрыть ошибку от пользователя.
В) Для увеличения объёма документации.
Г) Для автоматического исправления дефектов.

10. Какой инструмент чаще всего используют для ручного тестирования веб-интерфейсов и проверки API?
А) JaCoCo.
Б) Postman.
В) pytest.
Г) JMeter.

11. Какая метрика характеризует количество успешных запросов в секунду?
А) RPS (Requests Per Second).
Б) Latency.
В) Error rate.
Г) Throughput (в байтах).

12. Что проверяет стресс-тестирование?
А) Поведение системы при нагрузках, превышающих нормальные, и её способность восстанавливаться.
Б) Только корректность функционала.
В) Только время отклика при нормальной нагрузке.
Г) Безопасность системы.

13. Какой инструмент применяется для нагрузочного тестирования?
А) pytest.
Б) Git.
В) Apache JMeter.
Г) Postman (только для отдельных запросов, не для нагрузки).

14. Какой стандарт описывает характеристики и модели качества ПО?
А) ГОСТ Р 7.0.108-2022.
Б) ISO/IEC 25000.
В) IEEE 829.
Г) ISO 9001.

15. Что такое регрессионное тестирование?
А) Повторное выполнение ранее пройденных тестов для проверки, что новые изменения не сломали существующий функционал.
Б) Тестирование новой функциональности.
В) Тестирование производительности.
Г) Тестирование безопасности.