

Федеральное государственное образовательное бюджетное учреждение высшего образования
«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»
(Финансовый университет)

Кафедра информационных технологий
Факультет информационных технологий и анализа больших данных

Документ подписан усиленной неквалифицированной электронной подписью.

Организация: «ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ
РОССИЙСКОЙ ФЕДЕРАЦИИ»

Сертификат: otTv/dDRhDthxn+ux8/N/7BCdMflojMN

Утверждено: Проректор по учебной и методической работе, Е.А. Каменева

Дата: 10.02.2026 г.

О. Алхажа

Тестирование программного обеспечения

Рабочая программа дисциплины

для студентов, обучающихся по направлению подготовки

01.03.02 - Прикладная математика и информатика,

Образовательная программа «Прикладное машинное обучение»

Профиль:

«Прикладное машинное обучение»

Рекомендовано

*Факультет информационных технологий и анализа больших данных
(протокол № 07 от 22.04.2026 г.)*

Одобрено

*Кафедра информационных технологий
(протокол № 04 от 13.04.2026 г.)*



Содержание

1. Наименование дисциплины	3
2. Перечень планируемых результатов освоения образовательной программы (перечень компетенций) с указанием индикаторов их достижения и планируемых результатов обучения по дисциплине	3
3. Место дисциплины в структуре образовательной программы	
4. Объем дисциплины (модуля) в зачетных единицах и в академических часах с выделением объема аудиторной (лекции, семинары) и самостоятельной работы обучающихся	5
5. Содержание дисциплины, структурированное по темам (разделам) дисциплины с указанием их объемов (в академических часах) и видов учебных занятий	6
5.1. Содержание дисциплины	6
5.2. Учебно – тематический план	8
5.3. Содержание семинаров, практических занятий	10
6. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине	13
6.1. Перечень вопросов, отводимых на самостоятельное освоение дисциплины, формы внеаудиторной самостоятельной работы	13
6.2. Перечень вопросов, заданий, тем для подготовки к текущему контролю	16
7. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине	17
8. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины	7
9. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины	7
10. Методические указания для обучающихся по освоению дисциплины	30
11. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)	31
12. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине	31



1. Наименование дисциплины

Тестирование программного обеспечения

2. Перечень планируемых результатов освоения образовательной программы (перечень компетенций) с указанием индикаторов их достижения и планируемых результатов обучения по дисциплине

Код компетенции	Наименование компетенции	Индикаторы достижения компетенции	Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции
ПКП-7	Способность создавать прикладные программные компоненты и интегрировать в них методы анализа данных, машинного обучения и искусственного интеллекта	Владеет инструментальными средствами разработки прикладных программных продуктов: языками программирования, библиотеками, фреймворками	Знать: синтаксис и возможности языков (Python, Java, C++); библиотеки для анализа данных (NumPy, pandas, Polars); библиотеки ML (scikit-learn, XGBoost, LightGBM); фреймворки глубокого обучения (TensorFlow, PyTorch); среду разработки (Jupyter, VS Code, PyCharm); инструменты управления зависимостями (pip, conda, Poetry); Уметь: писать эффективный код на Python; обрабатывать данные через pandas/NumPy; строить ML-пайплайны через Pipeline; создавать и обучать нейросети в TensorFlow/PyTorch; настраивать изолированные окружения (venv/conda);
		Владеет архитектурными принципами и паттернами создания прикладных программных продуктов на различных платформах	Знать: паттерны ML-систем (пакетная/поточная обработка); микросервисную архитектуру для ML; способы развёртывания моделей (REST API, gRPC, batch); концепции feature store; оркестрацию пайплайнов (Airflow, Kubeflow); распределённые вычисления (Spark, Dask, Ray); Уметь: проектировать ETL/ELT для признаков; создавать



Код компетенции	Наименование компетенции	Индикаторы достижения компетенции	Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции
			микросервис инференса (FastAPI + Docker); применять feature store (Feast, Hopsworks); строить пайплайны обучения в Airflow; оптимизировать инференс (ONNX, TensorRT);
		Демонстрирует понимание побочных процессов, сопровождающих прикладную разработку программного обеспечения в промышленных условиях и владеет соответствующим инструментарием	Знать: CI/CD для ML (MLOps); логирование экспериментов (MLflow, W&B, TensorBoard); версионирование данных и моделей (DVC, Git LFS); мониторинг дрейфа (Evidently AI, NannyML); воспроизводимость экспериментов; контейнеризацию и оркестрацию (Docker, Kubernetes); Уметь: настраивать CI/CD для переобучения и деплоя; логировать параметры и метрики через MLflow; версионировать данные с DVC; настраивать дашборд дрейфа с оповещениями; разворачивать сервис инференса в Kubernetes; анализировать качество модели в продакшене;



3. Место дисциплины в структуре образовательной программы

Дисциплина «Тестирование программного обеспечения» относится к «Модулю "Архитектура программного обеспечения"».

4. Объем дисциплины (модуля) в зачетных единицах и в академических часах с выделением объема аудиторной (лекции, семинары) и самостоятельной работы обучающихся

Очная форма обучения

Вид учебной работы по дисциплине	Всего (в з/е и часах)	Семестр 7 (в часах)
Общая трудоемкость дисциплины	3/108	108
<i>Контактная работа – Аудиторные занятия</i>	<i>50</i>	<i>50</i>
<i>Лекции</i>	<i>16</i>	<i>16</i>
<i>Семинары, практические занятия</i>	<i>34</i>	<i>34</i>
<i>Самостоятельная работа</i>	<i>58</i>	<i>58</i>
Вид текущего контроля	Контрольная работа	Контрольная работа
Вид промежуточной аттестации	Зачет	Зачет



5. Содержание дисциплины, структурированное по темам (разделам) дисциплины с указанием их объемов (в академических часах) и видов учебных занятий

5.1. Содержание дисциплины

Тема 1. Основы тестирования программного обеспечения

Понятие качества ПО. Виды и уровни тестирования: функциональное, нефункциональное, модульное, интеграционное, системное, приемочное. Жизненный цикл тестирования. Дефекты: классификация, жизненный цикл, атрибуты. Тестовая документация: тест-план, тест-кейс, чек-лист, баг-репорт. Роль тестирования в процессах разработки (Waterfall, Agile, DevOps).

Тема 2. Функциональное тестирование. Техники тест-дизайна

Анализ требований, составление тест-плана. Техники тест-дизайна: классы эквивалентности (EP), анализ граничных значений (BVA), таблицы решений (Decision Tables), попарное тестирование. Разработка тест-кейсов. Позитивное и негативное тестирование. Примеры на основе реальных приложений.

Тема 3. Модульное тестирование и автоматизация с pytest

Принципы модульного тестирования. Фреймворк pytest: фикстуры, параметризация, проверка исключений. Оценка покрытия кода (coverage). Рефакторинг и добавление новой функциональности под тестами.

Тема 4. Нефункциональное тестирование: производительность, безопасность, удобство использования

Тестирование производительности: нагрузочное, стресс-тестирование, инструменты Apache JMeter. Метрики (время отклика, пропускная способность, ошибки). Тестирование безопасности: OWASP ZAP, анализ уязвимостей (SQL-инъекции, XSS, CSRF). Тестирование удобства использования (эвристики Нильсена, доступность WCAG).

Тема 5. Углубленное нефункциональное тестирование: надежность, масштабируемость, восстановление

Тестирование отказоустойчивости (Chaos Engineering, эмуляция сбоев). Тестирование масштабируемости (горизонтальное масштабирование, auto-scaling).



Интеграционное тестирование API (retry, circuit breaker). Тестирование восстановления (RTO/RPO, целостность данных).

Тема 6. Автоматизация тестирования веб-интерфейсов

Selenium WebDriver: установка, работа с браузером, поиск элементов, ожидания. Паттерн Page Object Model. Параллельный запуск тестов. Allure Framework для генерации отчетов.

Тема 7. CI/CD и автоматизация запуска тестов

Понятие непрерывной интеграции и доставки. Настройка пайплайна в GitLab CI (GitVerse). Автоматический запуск тестов при изменениях кода, сборка артефактов, уведомления.

Тема 8. Тестирование ETL-пайплайнов

Понятие ETL (Extract, Transform, Load). Тестирование извлечения, преобразования, загрузки данных. Использование Pandas, SQLite. Проверка качества данных: полнота, точность, консистентность. Great Expectations для валидации.

Тема 9. Качество данных. Data Quality Framework

Мониторинг качества данных: метрики, алерты, визуализация. Разработка фреймворка для проверки полноты, точности, дрейфа данных. Создание дашборда на Flask. Система уведомлений (email, консоль).

Тема 10. Тестирование облачных сервисов

Облачные data-сервисы (S3, SQS). Локальная эмуляция с LocalStack. Разработка клиента на boto3. Тестирование операций с хранилищем и очередями. Создание облачного пайплайна.

Тема 11. Chaos Engineering для data-систем

Принципы Chaos Engineering. Создание Chaos Framework: сетевые задержки, отказ сервисов, высокая нагрузка на CPU/память, коррупция данных. Устойчивый пайплайн с retry и circuit breaker. Мониторинг устойчивости, отчеты.

Тема 12. Комплексное тестирование ML-пайплайнов

Жизненный цикл ML-проекта. Тестирование данных (качество, дрейф), тестирование модели (метрики, fairness), тестирование API инференса. Мониторинг ML-систем (Evidently). Интеграция в CI/CD.



5.2. Учебно – тематический план

Очная форма обучения

п/ п	Наименование тем (разделов) дисциплины	Трудоемкость в часах				
		Всего	Контактная работа - Аудиторная работа			Самостоятельная работа
			Общая, в т.ч.:	Лекции	Семинары, практические занятия	
1	Основы тестирования программного обеспечения	7	3	1	2	4
2	Функциональное тестирование. Техники тест-дизайна	10	6	2	4	4
3	Модульное тестирование и автоматизация с pytest	8	4	2	2	4
4	Нефункциональное тестирование: производительность, безопасность, удобство использования	10	6	2	4	4
5	Углубленное нефункциональное тестирование: надежность, масштабируемость, восстановление	10	6	2	4	4
6	Автоматизация тестирования веб-интерфейсов	8	4	1	3	4
7	CI/CD и автоматизация запуска тестов	7	3	1	2	4



п/ п	Наименование тем (разделов) дисциплины	Трудоемкость в часах				
		Всего	Контактная работа - Аудиторная работа			Самостоятельная работа
			Общая, в т.ч.:	Лекции	Семинары, практические занятия	
8	Тестирование ETL- пайплайнов	8	4	1	3	4
9	Качество данных. Data Quality Framework	8	3	1	2	5
10	Тестирование облачных сервисов	8	3	1	2	5
11	Chaos Engineering для data- систем	12	5	1	4	7
12	Комплексное тестирование ML-пайплайнов	12	3	1	2	9
В целом по дисциплине		108	50	16	34	58



5.3. Содержание семинаров, практических занятий

Наименование тем (разделов) дисциплины	Перечень вопросов для обсуждения на семинарах, практических занятиях	Формы проведения занятий
Основы тестирования программного обеспечения	1. Что такое дефект, как его описывать? 2. Виды тестирования (ручное/автоматизированное, функциональное/нефункциональное). 3. Структура баг-репорта.	Выполнение практического задания: исследовательское тестирование выбранного веб-сайта, поиск и оформление минимум двух багов. Оформление отчета.
Функциональное тестирование. Техники тест-дизайна	1. Анализ требований, составление тест-плана. 2. Техники EP, BVA, таблицы решений. 3. Разработка тест-кейсов для калькулятора (сложение). 4. Разработка тестовой документации для веб-формы (SkyScanner).	Создание тест-кейсов для калькулятора (не менее 5), практические занятия.
Модульное тестирование и автоматизация с pytest	1. Установка Python, pytest. 2. Написание тестов для функции. 3. Параметризация, проверка исключений. 4. Оценка покрытия кода.	Практическое занятие.
Нефункциональное тестирование: производительность, безопасность, удобство использования	1. Нагрузочное тестирование в JMeter. 2. Сканирование безопасности в OWASP ZAP. 3. Эвристическая оценка UX и доступности.	Практическое занятие.



Наименование тем (разделов) дисциплины	Перечень вопросов для обсуждения на семинарах, практических занятиях	Формы проведения занятий
Углубленное нефункциональное тестирование: надежность, масштабируемость, восстановление	1. Тестирование отказоустойчивости (эмуляция сбоев). 2. Тестирование масштабируемости. 3. API reliability и восстановление.	Практическое занятие.
Автоматизация тестирования веб-интерфейсов	1. Настройка Selenium WebDriver. 2. Паттерн Page Object Model. 3. Параллельный запуск, отчеты Allure.	Практическое занятие.
CI/CD и автоматизация запуска тестов	1. Понятие CI/CD. 2. Настройка пайплайна в GitVerse (GitLab CI). 3. Артефакты и уведомления.	Практическое занятие.
Тестирование ETL-пайплайнов	1. ETL-процесс. 2. Тестирование извлечения, преобразования, загрузки. 3. Использование Pandas, SQLite.	Практическое занятие.
Качество данных. Data Quality Framework	1. Метрики качества данных. 2. Фреймворк для проверки полноты, точности, консистентности. 3. Визуализация и дашборд.	Практическое занятие.
Тестирование облачных сервисов	1. Эмуляция AWS через LocalStack. 2. Работа с S3, SQS через boto3. 3. Интеграционное тестирование.	Практическое занятие.



Наименование тем (разделов) дисциплины	Перечень вопросов для обсуждения на семинарах, практических занятиях	Формы проведения занятий
Chaos Engineering для data-систем	1. Принципы Chaos Engineering. 2. Создание Chaos Framework. 3. Устойчивый пайплайн (retry, circuit breaker). 4. Мониторинг устойчивости.	Практическое занятие.
Комплексное тестирование ML-пайплайнов	1. Тестирование данных и модели. 2. Тестирование API инференса. 3. Мониторинг ML-систем.	Практическое занятие.



6. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине

6.1. Перечень вопросов, отводимых на самостоятельное освоение дисциплины, формы внеаудиторной самостоятельной работы

Наименование тем (разделов) дисциплины	Перечень вопросов, отводимых на самостоятельное освоение	Формы внеаудиторной самостоятельной работы
Основы тестирования программного обеспечения	Понятие качества по ISO 25010. Процессы тестирования в Agile и DevOps.	Изучение литературы, подготовка к семинару.
Функциональное тестирование. Техники тест-дизайна	Дополнительные техники тест-дизайна: попарное тестирование, попарное комбинирование.	Выполнение лабораторных работ; оформление отчетов.
Модульное тестирование и автоматизация с pytest	Альтернативные фреймворки: unittest, doctest. Профилирование кода.	Выполнение лабораторной работы; изучение документации pytest.
Нефункциональное тестирование: производительность, безопасность, удобство использования	Метрики производительности, профилирование приложений. OWASP Top 10.	Выполнение лабораторной работы; подготовка отчета.
Углубленное нефункциональное тестирование: надежность, масштабируемость, восстановление	Практики Site Reliability Engineering (SRE). Инструменты для Chaos Engineering (Chaos Mesh).	Выполнение лабораторной работы; изучение кейсов.
Автоматизация тестирования веб-	WebDriverWait, Expected Conditions. Интеграция с CI/CD.	Выполнение лабораторной работы; оформление отчетов.



Наименование тем (разделов) дисциплины	Перечень вопросов, отводимых на самостоятельное освоение	Формы внеаудиторной самостоятельной работы
интерфейсов		
CI/CD и автоматизация запуска тестов	Другие CI/CD системы (Jenkins, GitHub Actions).	Выполнение лабораторной работы; создание пайплайна в GitVerse.
Тестирование ETL- пайплайнов	Тестирование производительности ETL, параллельная загрузка.	Выполнение лабораторной работы; написание тестов.
Качество данных. Data Quality Framework	Продвинутое методы мониторинга качества данных (Data Observability).	Выполнение лабораторной работы; создание дашборда.
Тестирование облачных сервисов	Реальные облачные провайдеры (AWS, Yandex Cloud).	Выполнение лабораторной работы; тестирование с LocalStack.
Chaos Engineering для data- систем	Экономическая эффективность Chaos Engineering.	Выполнение лабораторной работы; анализ отчетов.
Комплексное тестирование ML-пайплайнов	MLOps: CI/CD/CT для ML. Инструменты мониторинга (Evidently, WhyLogs).	Выполнение лабораторной работы; подготовка к защите проекта.



6.2. Перечень вопросов, заданий, тем для подготовки к текущему контролю

Примерные вопросы для подготовки к контрольной работе

1. Перечислите уровни тестирования по V-модели. Дайте характеристику каждому.
2. Что такое тест-кейс? Какие поля он содержит? Приведите пример.
3. Техники тест-дизайна: классы эквивалентности и граничные значения. Приведите примеры.
4. В чем отличие позитивного тестирования от негативного?
5. Что такое модульный тест? Преимущества автоматизированного модульного тестирования.
6. Опишите процесс нагрузочного тестирования в JMeter. Какие метрики собираются?
7. Какие виды уязвимостей проверяет OWASP ZAP? Приведите примеры.
8. Что такое Page Object Model? Зачем он нужен?
9. Как настроить CI/CD-пайплайн для автоматического запуска тестов?
10. Какие этапы включает тестирование ETL-пайплайна?
11. Перечислите метрики качества данных. Как их проверить с помощью Great Expectations?
12. Что такое LocalStack? Для каких целей он применяется?
13. Принципы Chaos Engineering. Приведите примеры экспериментов.
14. Какие аспекты тестируются в ML- пайплайне? Назовите инструменты для каждого.

Примерные задания контрольной работы

1. Разработайте тест-план и набор тест-кейсов для функции расчета скидки (по заданным условиям). Примените техники EP и BVA.
2. Напишите модульные тесты на pytest для функции, проверяющей корректность ввода email-адреса. Используйте параметризацию.
3. Составьте чек-лист для тестирования формы регистрации на сайте. Укажите не менее 10 пунктов.
4. Опишите сценарий нагрузочного тестирования для REST API, укажите инструменты и ожидаемые метрики.
5. Предложите архитектуру Data Quality Framework для мониторинга таблицы с клиентами. Какие проверки должны быть?



Дополнительная информация

Критерии балльной оценки различных форм текущего контроля успеваемости содержатся в соответствующих методических рекомендациях Кафедры информационных технологий Факультета информационных технологий и анализа больших данных.



7. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине

Перечень компетенций с указанием индикаторов их достижения в процессе освоения образовательной программы содержится в разделе 2. «Перечень планируемых результатов освоения образовательной программы (перечень компетенций) с указанием индикаторов их достижения и планируемых результатов обучения по дисциплине».

Типовые контрольные задания или иные материалы, необходимые для оценки индикаторов достижения компетенций, умений и знаний

Наименование компетенции	Наименование индикаторов достижения компетенции	Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции	Типовые контрольные задания
ПКП-7. Способность создавать прикладные программные компоненты и интегрировать в них методы анализа данных, машинного обучения и искусствен-	Владеет инструментальными средствами разработки прикладных программных продуктов: языками программирования, библиотеками, фреймворками	Знать: синтаксис и возможности языков (Python, Java, C++); библиотеки для анализа данных (NumPy, pandas, Polars); библиотеки ML (scikit-learn, XGBoost, LightGBM); фреймворки глубокого обучения (TensorFlow, PyTorch); среду разработки (Jupyter, VS	1. Анализ требований и разработка тестовой документации. По заданному техническому заданию (например, модуль «Калькулятор комиссии» из ЛР №1) выполнить: - составить список неоднозначностей и уточняющих вопросов (не менее 5); - разработать тест-план (объект, цели, критерии начала/окончания, подходы); - создать не менее 12 тест-кейсов, покрывающих все граничные значения и классы эквивалентности, включая негативные проверки. 2. Документирование дефектов. На основе исследовательского тестирования веб-приложения (семинар 1) оформить минимум 2 баг-репорта по стандартному шаблону (заголовок, шаги воспроизведения, ожидаемый/фактический результат, серьезность, скриншот).



Наименование компетенции	Наименование индикаторов достижения компетенции	Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции	Типовые контрольные задания
ного интеллекта		Code, PyCharm); инструменты управления зависимостями (pip, conda, Poetry); Уметь: писать эффективный код на Python; обрабатывать данные через pandas/NumPy; строить ML-пайплайны через Pipeline; создавать и обучать нейросети в TensorFlow/PyTorch; настраивать изолированные окружения (venv/conda);	
	Владеет архитектурными принципами и паттернами создания прикладных про-	Знать: паттерны ML-систем (пакетная/поточная обработка); микросервисную ар-	1. Модульное тестирование (pytest). Разработать модульные тесты для функции расчета комиссии (ЛР №3). Использовать параметризацию, проверку исключений. Обеспечить покрытие кода 100%, предоставить отчет о покрытии.



Наименование компетенции	Наименование индикаторов достижения компетенции	Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции	Типовые контрольные задания
	граммных продуктов на различных платформах	хитектуру для ML; способы развёртывания моделей (REST API, gRPC, batch); концепции feature store; оркестрацию пайплайнов (Airflow, Kubeflow); распределённые вычисления (Spark, Dask, Ray); Уметь: проектировать ETL/ELT для признаков; создавать микросервис inference (FastAPI + Docker); применять feature store (Feast, Hopsworks); строить пайплайны обучения в Airflow; оптимизи-	2. Нагрузочное тестирование (JMeter). Провести нагрузочное тестирование указанного веб-сайта (ЛР №4): создать сценарий постепенного увеличения нагрузки, пиковой нагрузки, длительной нагрузки. Построить графики времени отклика, пропускной способности, проанализировать ошибки. Предоставить конфигурацию JMeter и отчет с выводами. 3. Автоматизация веб-тестирования (Selenium). Разработать автоматизированные тесты для формы поиска Яндекс (ЛР №6). Реализовать паттерн Page Object Model, использовать параметризацию, параллельный запуск. Сгенерировать отчет Allure. Код разместить в репозитории GitVerse. 4. CI/CD пайплайн. Создать в GitVerse проект с кодом тестов из ЛР №6. Настроить .gitlab-ci.yml для автоматического запуска тестов при коммите в основную ветку. Предоставить скриншоты успешного выполнения пайплайна и логов. 5. Тестирование ETL и качества данных. Реализовать ETL-пайплайн (ЛР №8) с извлечением из CSV, преобразованиями и загрузкой в SQLite. Написать тесты с использованием Great Expectations (ЛР №9). Создать дашборд на Flask для визуализации метрик качества данных.



Наименование компетенции	Наименование индикаторов достижения компетенции	Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции	Типовые контрольные задания
		ровать инференс (ONNX, TensorRT);	
	Демонстрирует понимание побочных процессов, сопровождающих прикладную разработку программного обеспечения в промышленных условиях и владеет соответствующим инструментарием	Знать: CI/CD для ML (MLOps); логирование экспериментов (MLflow, W&B, TensorBoard); версионирование данных и моделей (DVC, Git LFS); мониторинг дрефта (Evidently AI, NannyML); воспроизводимость экспериментов; контейнеризацию и оркестрацию (Docker, Kubernetes); Уметь: настраивать CI/CD для переобучения и деплоя; логировать параметры и метрики через	1. Комплексное нефункциональное тестирование. Провести тестирование производительности, безопасности и удобства использования веб-приложения (ЛР №4, №5). Составить отчет, включающий: результаты нагрузочного тестирования (графики, анализ узких мест), результаты сканирования безопасности (OWASP ZAP), эвристическую оценку UX по критериям Нильсена. Предложить приоритизированные рекомендации. 2. Chaos Engineering. Создать Chaos Framework (ЛР №11) для data-системы: реализовать эксперименты (сетевая задержка, отказ S3, высокая нагрузка CPU, коррупция данных). Построить устойчивый пайплайн с retry и circuit breaker. Провести серию экспериментов, оценить успешность пайплайна. Сформировать отчет с графиками и выводами об устойчивости системы. 3. Тестирование ML-пайплайна. Разработать ML-пайплайн (генерация данных, обучение модели, API) (ЛР №12). Выполнить: - тестирование качества данных (полнота, выбросы, дрефт); - тестирование модели (метрики accuracy, precision, recall, fairness); - тестирование API (health, predict, batch_predict);



Наименование компетенции	Наименование индикаторов достижения компетенции	Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции	Типовые контрольные задания
		MLflow; версионировать данные с DVC; настраивать дашборд дрефта с оповещениями; разворачивать сервис инференса в Kubernetes; анализировать качество модели в продакшене;	- настройку мониторинга дрефта данных и модели с использованием Evidently. Предоставить код, отчеты тестирования и мониторинга, скриншоты дашборда. 4. Контрольная работа (комплексное задание). Выполнить индивидуальное задание, включающее элементы всех трех индикаторов: анализ требований, разработка тест-кейсов, написание модульных тестов, автоматизация сценария в Selenium, настройка CI/CD и анализ нефункциональных характеристик.



Примеры практико-ориентированных заданий

А. Ручное тестирование (тест-кейсы, баг-репорты, чек-листы)

1. Создайте 4 тест-кейса для проверки функции "Расчет комиссии за перевод" (сумма от 100 до 50000 руб.) с использованием техники граничных значений.
2. Рассчитайте классы эквивалентности для поля "Возраст пользователя" (0-120 лет). Создайте 6 тест-кейсов: 3 позитивных, 3 негативных.
3. Создайте тест-план для модуля "Авторизация пользователя". Включите: цель, критерии начала/окончания, подходы (минимум 3), риски.
4. Составьте баг-репорт для ошибки: "Форма регистрации принимает email без @". Используйте шаблон: ID, заголовок, шаги, результат, severity.
5. Составьте чек-лист из 5 пунктов для проверки мобильной версии сайта (адаптивность, шрифты, кнопки).
6. Создайте тест-комплект из 5 тест-кейсов для проверки сортировки товаров (по цене, названию, рейтингу, возрастание/убывание, по умолчанию).
7. Составьте тест-план для проверки фильтров на Wildberries (цена, производитель, цвет). Укажите риски и приоритеты.
8. Используя технику таблицы решений, составьте тест-кейсы для функции "Скидка пользователя" (Возраст, Сумма покупок, Статус).
9. Составьте 8 тест-кейсов для проверки системы отзывов на товар: добавление, редактирование, удаление, оценка, комментарий.
10. Составьте 6 тест-кейсов для проверки формы "Оставить отзыв" (имя, email, текст, рейтинг). Включите пустые поля и спецсимволы.
11. Составьте тест-план для нагрузочного тестирования корзины интернет-магазина: цели, сценарии (10, 100, 1000 пользователей), метрики.
12. Создайте 5 тест-кейсов для проверки подписки на рассылку: валидный email, невалидный, повторная подписка, отписка.
13. Реализуйте тестовый фреймворк для проверки поиска на demo.opencart.com: 3 Page Object (MainPage, SearchPage, ProductPage) и 2 теста.
14. Составьте 7 тест-кейсов для проверки экспорта данных в форматах CSV, Excel, PDF. Укажите проверки для каждого формата.
15. Создайте тест-план для проверки интеграции с платежным шлюзом (success, failure, timeout, refund).
16. Составьте 5 тест-кейсов для проверки API калькулятора (используйте reqres.in или JSONPlaceholder). Укажите метод, URL, ожидаемый статус.
17. Разработайте стратегию тестирования для системы лояльности: начисление баллов, списание, сгорание, повышение уровня.



18. Составьте 6 тест-кейсов для проверки системы сравнения товаров: добавление, удаление, очистка, лимит, публичность.
19. Составьте 5 тест-кейсов для проверки системы логирования: уровни логов, формат, ротация, хранение, анализ.
20. Составьте итоговый экзаменационный вопрос: Опишите полный процесс тестирования новой функции "Онлайн-чат с поддержкой" от начала до конца.
21. Составьте 4 тест-кейса для проверки логина через соцсети.
22. Составьте баг-репорт для несовместимости с браузером Safari.
23. Напишите баг-репорт для ошибки входа с неправильным паролем на любом сайте (структура: заголовок, шаги, ожидаемый/фактический результат).
24. Создайте 3 тест-кейса для проверки корзины покупок (добавление, удаление, изменение количества).
25. Напишите 2 позитивных и 1 негативный тест-кейс для формы регистрации с email.
26. Составьте тест-комплект (test suite) из 3 тестов для авторизации.
27. Напишите 3 тест-кейса для сортировки товаров по цене (возрастание, убывание, по умолчанию).
28. Составьте чек-лист для проверки мобильной версии сайта (5 пунктов).
29. Напишите 2 тест-кейса для проверки фильтров (производитель, цена).
30. Составьте баг-репорт для ошибки 404 на несуществующей странице.
31. Напишите 3 тест-кейса для проверки валидации формы (пустое поле, слишком длинное значение, спецсимволы).
32. Составьте баг-репорт для медленной загрузки страницы (больше 5 секунд).
33. Напишите 2 позитивных тест-кейса для поиска (точный запрос, частичный запрос).
34. Напишите 3 тест-кейса для проверки навигационного меню.
35. Составьте чек-лист для проверки безопасности сайта (HTTPS, куки, пароль).
36. Напишите 3 тест-кейса для проверки подписки на новости.
37. Составьте баг-репорт для отсутствующей кнопки "купить" на мобильной версии.
38. Напишите 2 тест-кейса для проверки сравнения товаров.
39. Напишите 3 тест-кейса для проверки работы купона на скидку.
40. Составьте чек-лист для проверки доступности сайта (5 пунктов).

Б. Автоматизация (Jupyter Notebook + Python + Selenium)

1. Напишите код в Jupyter для теста калькулятора: проверьте, что $2+2=4$ и выведите результат в формате "Тест <номер>: [PASS/FAIL]".



2. В Jupyter откройте <https://ya.ru>, выполните поиск, проверьте наличие рекламы в результатах.
3. В Jupyter откройте <https://ya.ru>, найдите поле поиска по ID, введите "тестирование" и проверьте, что появились подсказки.
4. В Jupyter откройте <https://demo.opencart.com>, найдите элемент корзины по классу, выведите текст корзины.
5. В Jupyter откройте любой сайт, сделайте скриншот, сохраните его в файл.
6. В Jupyter откройте <https://demo.opencart.com>, найдите ссылку "My Account", выведите её href.
7. В Jupyter откройте <https://ya.ru>, найдите логотип по тегу, выведите его атрибут src.
8. В Jupyter откройте <https://demo.opencart.com>, перейдите в раздел "Phones", выведите количество товаров.
9. В Jupyter откройте <https://ya.ru>, измените размер окна браузера на 800x600, сделайте скриншот.
10. В Jupyter откройте <https://demo.opencart.com>, найдите цену первого товара, выведите её как число.
11. В Jupyter откройте <https://ya.ru>, найдите ссылку на "Картинки", кликните по ней.
12. В Jupyter откройте <https://demo.opencart.com>, добавьте товар в корзину через кнопку, выведите сообщение об успехе.
13. В Jupyter откройте <https://ya.ru>, проверьте, что title страницы содержит "Яндекс".
14. В Jupyter откройте <https://demo.opencart.com>, найдите все ссылки в футере, выведите их количество.
15. В Jupyter откройте <https://ya.ru>, найдите элемент по CSS классу, выведите его текст.
16. В Jupyter откройте <https://demo.opencart.com>, переключитесь на валюту USD, выведите цену в долларах.
17. В Jupyter откройте <https://ya.ru>, найдите кнопку "Найти", проверьте её текст.
18. В Jupyter откройте <https://demo.opencart.com>, найдите элемент по XPath, выведите его tag.
19. В Jupyter откройте <https://ya.ru>, выполните поиск, извлеките URL первой ссылки.
20. В Jupyter откройте <https://demo.opencart.com>, проверьте, что страница загрузилась за <5 секунд.
21. В Jupyter откройте <https://ya.ru>, найдите все изображения на странице, выведите их количество.



22. В Jupyter откройте <https://demo.opencart.com>, найдите кнопку "Add to Cart", проверьте её цвет через CSS.
23. В Jupyter откройте <https://ya.ru>, выполните JavaScript: `document.title = "Тест"` и проверьте изменение.
24. В Jupyter откройте <https://demo.opencart.com>, найдите элемент по частичному тексту, выведите его полный текст.
25. В Jupyter откройте <https://ya.ru>, проверьте, что страница содержит куки с именем "yandexuid".
26. В Jupyter откройте <https://demo.opencart.com>, найдите элемент, прокрутите страницу до него, сделайте скриншот.
27. В Jupyter откройте <https://ya.ru>, выполните поиск, подождите 2 секунды, извлеките текст результата.
28. В Jupyter откройте <https://demo.opencart.com>, проверьте, что все картинки имеют атрибут alt.
29. В Jupyter откройте <https://ya.ru>, найдите ссылки в подвале, выведите их текст.
30. В Jupyter откройте <https://demo.opencart.com>, найдите элемент, измените его текст через JS, проверьте.
31. В Jupyter откройте <https://ya.ru>, выполните поиск, проверьте количество результатов > 0 .
32. В Jupyter откройте <https://demo.opencart.com>, найдите цену товара, выведите её как float.
33. В Jupyter откройте <https://ya.ru>, найдите кнопку, проверьте, что она кликабельна.
34. В Jupyter откройте <https://demo.opencart.com>, найдите все кнопки, выведите их классы.
35. В Jupyter откройте <https://ya.ru>, выполните поиск, извлеките все заголовки результатов h3.
36. В Jupyter откройте <https://demo.opencart.com>, измените язык, проверьте текст корзины.
37. В Jupyter откройте <https://ya.ru>, найдите favicon, выведите её размер.
38. В Jupyter откройте <https://demo.opencart.com>, найдите элемент по XPath, проверьте его видимость.
39. В Jupyter откройте <https://ya.ru>, выполните поиск "Python 3.12", перейдите на первый результат.
40. В Jupyter используйте Selenium для открытия <https://gitverse.ru>, найдите поле поиска, введите "python", выполните поиск и выведите количество найденных репозиторий.



Примеры вопросов для подготовки к промежуточной аттестации

Примерные вопросы для подготовки к зачету

1. Назовите 3 принципа тестирования и приведите примеры их нарушения.
2. Чем отличается ошибка (error) от дефекта (defect) и сбоя (failure)?
3. Опишите роли тестировщика (аналитик, адвокат пользователя). Какие качества важны?
4. Расскажите о принципе "Раннее тестирование". Почему он экономит деньги?
5. Что такое классы эквивалентности? Приведите пример для поля возраста (0-120).
6. Объясните принцип "Парадокс пестицида". Как его преодолеть?
7. Что такое CI/CD и какую роль играют в нем автоматизированные тесты?
8. Расскажите о Page Object Model. В чем его преимущества?
9. Чем отличается функциональное тестирование от нефункционального?
10. Опишите процесс тестирования (планирование, дизайн, выполнение, отчетность).
11. Расскажите о тестировании API (REST). Какие методы HTTP используются?
12. Что такое XSS и SQL-инъекция? Как защититься?
13. Опишите технику граничных значений для поля "количество" (1-100).
14. Расскажите о параллельном выполнении тестов. Зачем оно нужно?
15. Что такое регрессионное тестирование? Когда его проводят?
16. Расскажите о тестировании мобильных приложений (Appium, Espresso).
17. Опишите процесс нагрузочного тестирования. Какие инструменты знаете?
18. Что такое Docker и как он помогает в тестировании?
19. Расскажите о тестировании в микросервисной архитектуре.
20. Что такое A/B тестирование? Приведите пример.
21. Расскажите о паттерне Page Factory (Java).
22. Опишите процесс ревью тестовой документации.
23. Что такое тестовое покрытие (code coverage)? Как его измерить?
24. Расскажите о безопасном хранении паролей в автотестах.
25. Что такое "самовосстанавливающиеся тесты" (self-healing)?
26. Опишите разницу между TDD и BDD.
27. Расскажите о тестировании с помощью mock-объектов.
28. Что такое "тестирование как код" (Testing as Code)?
29. Опишите процесс настройки тестового окружения.
30. Расскажите о роли QA в DevOps культуре.
31. Что такое chaos engineering и как тестировать отказоустойчивость?
32. Опишите процесс тестирования ML-модели.



33. Расскажите о метриках качества автоматизированных тестов.
34. Что такое "shift-left testing" и "shift-right testing"?
35. Опишите процесс performance testing для веб-приложения.
36. Расскажите о сертификациях в области тестирования (ISTQB, AT*).
37. Что такое контрактное тестирование (Contract Testing)?
38. Опишите процесс тестирования безопасности (security testing).
39. Расскажите о будущем тестирования: AI, low-code, self-healing.
40. Что такое "тестовый дизайн" и какие техники вы знаете?



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

Основная литература:

1. Хруцкий, Валерий Евгеньевич Оценка персонала. Сбалансированная система показателей : практическое пособие (отсутствует) / В. Е. Хруцкий, Р. А. Толмачев, Р. В. Хруцкий. 3-е изд., испр. и доп Электрон. дан. Москва : Юрайт, 2026 203 с (Профессиональная практика) URL: <https://urait.ru/bcode/585679> (дата обращения: 27.02.2026). Режим доступа: Электронно-библиотечная система Юрайт, для авториз. пользователей <https://urait.ru/bcode/585679> ISBN 978-5-534-16778-8 : 1169.00. [БИК ID: RU2fURAIT2f585679]
2. Лаврищева, Екатерина Михайловна Программная инженерия. Парадигмы, технологии и CASE-средства : учебник для вузов / Е. М. Лаврищева. 2-е изд. Электрон. дан. Москва : Юрайт, 2026 280 с (Высшее образование) URL: <https://urait.ru/bcode/584533> (дата обращения: 27.02.2026). Режим доступа: Электронно-библиотечная система Юрайт, для авториз. пользователей <https://urait.ru/bcode/584533> ISBN 978-5-534-01056-5 : 1529.00. [БИК ID: RU2fURAIT2f584533]
3. Щербак, Алексей Викторович Тестирование программного обеспечения : учебник для вузов / А. В. Щербак. Электрон. дан. Москва : Юрайт, 2026 145 с (Высшее образование) URL: <https://urait.ru/bcode/590250> (дата обращения: 27.02.2026). Режим доступа: Электронно-библиотечная система Юрайт, для авториз. пользователей <https://urait.ru/bcode/590250> ISBN 978-5-534-19291-9 : 629.00. [БИК ID: RU2fURAIT2f590250]

Дополнительная литература:

1. Ехлаков, Ю. П. Управление программными проектами. Стандарты, модели [Электронный ресурс] / Ехлаков Ю. П. 3-е изд., стер. Санкт-Петербург : Лань, 2021 244 с. Книга из коллекции Лань - Информатика <https://e.lanbook.com/book/175498> ISBN 978-5-8114-8362-4. [БИК ID: RU-LAN-BOOK-175498]
2. Игнатьев, А. В. Тестирование программного обеспечения [Электронный ресурс] : учебное пособие для вузов / Игнатьев А. В. 4-е изд., стер. Санкт-Петербург : Лань, 2025 56 с. Книга из коллекции Лань - Информатика <https://e.lanbook.com/book/481331> ISBN 978-5-507-50858-7. [БИК ID: RU-LAN-BOOK-481331]



3. Пантелеев, Е. Р. Методы научных исследований в программной инженерии [Электронный ресурс] : учебное пособие для вузов / Пантелеев Е. Р. 2-е изд., стер. Санкт-Петербург : Лань, 2021 136 с. Книга из коллекции Лань - Информатика <https://e.lanbook.com/book/152439> ISBN 978-5-8114-6781-5. [БИК ID: RU-LAN-BOOK-152439]

Нормативные правовые акты:

-

9. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины

1. Документация pytest – <https://docs.pytest.org/>
2. OWASP ZAP – <https://www.zaproxy.org/>
3. Официальная документация Python – <https://docs.python.org/3/>
4. Selenium WebDriver – <https://www.selenium.dev/documentation/>
5. Allure Framework – <https://docs.qameta.io/allure/>
6. Apache JMeter – <https://jmeter.apache.org/>
7. GitLab CI/CD – <https://docs.gitlab.com/ee/ci/>
8. Great Expectations – <https://docs.greatexpectations.io/>
9. Evidently AI – <https://evidentlyai.com/>
10. GitVerse – <https://gitverse.ru/>
11. Портал Финансового университета – <https://fa.ru>



10. Методические указания для обучающихся по освоению дисциплины

Для успешного освоения дисциплины студентам необходимо:

1. Регулярно посещать лекции и вести конспект, фиксируя основные понятия, техники, инструменты.
2. Активно участвовать в семинарских занятиях, выполняя практические задания. Лабораторные работы выполняются индивидуально или в малых группах (по решению преподавателя). Отчеты по лабораторным работам должны быть оформлены в соответствии с требованиями и содержать:
 - Титульный лист;
 - Цель работы;
 - Ход выполнения (код, скриншоты, описание);
 - Выводы.
 - Исходный код должен быть размещен в репозитории GitVerse (GitLab) и доступен преподавателю.
3. Самостоятельно изучать теоретический материал, используя указанную литературу и интернет-ресурсы. Особое внимание уделить разделам, которые выносятся на самостоятельное освоение.
4. Подготовиться к контрольной работе, повторив вопросы из раздела 6.2., следует использовать нормативные документы Финансового университета, Методические рекомендации по планированию и организации внеаудиторной самостоятельной работы студентов по образовательным программам бакалавриата и магистратуры в Финансовом университете, утвержденные приказом Финуниверситета от 11.05.2021 г. № 1040/о (см. сайт Финансового Университета: на главной странице раздел "Наш университет" → "Единая правовая база Финуниверситета" → "Организация учебного процесса" → "Нормативные документы по самостоятельной работе").
5. Использовать современные инструменты для разработки и тестирования: Python, pytest, Selenium, JMeter, Git, Docker, LocalStack, Allure и др. Установка и настройка ПО описана в методических указаниях к лабораторным работам.
6. При выполнении лабораторных работ следовать пошаговым инструкциям, представленным в материалах. В случае затруднений обращаться к преподавателю на консультациях.
7. Соблюдать правила академической честности: не копировать работы других студентов; при использовании сторонних кодов указывать источники.



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

Комплект лицензионного программного обеспечения:

1. Python 3.8
2. Операционная система (Windows / macOS / Linux)
3. Docker Desktop Контейнеризация для локального развертывания документации
4. Среда разработки (PyCharm, VS Code или др.)
5. Apache JMeter
6. OWASP ZAP
7. Git (клиент)
8. Антивирус Kaspersky

Современные профессиональные базы данных и информационные справочные системы

1. eLIBRARY.RU
2. Электронная библиотека Финансового университета (<https://library.fa.ru>)
3. ЭБС Лань (<https://e.lanbook.com>) Доступ к учебной и научной литературе (Леоненков, Маклаков, Вигерс, Фаулер и др.)
4. ЭБС Юрайт (<https://urait.ru>) Доступ к учебникам и практикумам (Чистов, Черткова, Сергеев, Терехов и др.)
5. Использование системы управления версиями Git (репозиторий на GitVerse)
6. Применение средств автоматизации тестирования (pytest, Selenium)
7. Контейнеризация (Docker)
8. Организация CI/CD (GitLab CI)
9. Визуализация результатов тестирования (Allure, Matplotlib)

Сертифицированные программные и аппаратные средства защиты информации

1. не предусмотрены

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

1. **Учебная аудитория** для проведения учебных занятий, предусмотренных программой, в том числе групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, оснащенная



оборудованием и техническими средствами обучения: мебель аудиторная (столы, стулья, доска аудиторная), набор демонстрационного оборудования (проектор, экран)

2. **Компьютерный класс** для проведения учебных занятий, предусмотренных программой, в том числе групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, оснащенный оборудованием и техническими средствами обучения: мебель аудиторная (столы, стулья, доска аудиторная), персональные компьютеры, набор демонстрационного оборудования (проектор, экран)