МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»
(РУТ (МИИТ)

Рабочая программа дисциплины (модуля),
как компонент образовательной программы
базового высшего образования
по направлению подготовки
09.03.01 Информатика и вычислительная техника,
утвержденной директором ФГАОУ ВО РУТ
(МИИТ) Париновым Д.В.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Введение в IT

Направление подготовки: 09.03.01 Информатика и вычислительная техника

Направленность (профиль): Технологии разработки программного обеспечения

Форма обучения: Очная

Рабочая программа дисциплины (модуля) в виде электронного документа выгружена из единой корпоративной информационной системы управления университетом и соответствует оригиналу

Простая электронная подпись, выданная ФГАОУ ВО РУТ (МИИТ)
ID подписи: 937226
Подписал: директор центра Проневич Ольга Борисовна
Дата: 12.11.2026

1. Общие сведения о дисциплине (модуле).

Дисциплина «Введение в IT» является вводной общепрофессиональной дисциплиной первого семестра, которая формирует у студентов целостное представление о профессиональной роли инженера-программиста и о начальном цикле промышленной разработки программного обеспечения. Теоретический курс связывает основы информатики, архитектуры вычислительных систем, операционных и сетевых технологий с инженерией требований, алгоритмизацией на Python, управлением версиями, тестированием, рецензированием программного кода, непрерывной интеграцией, реляционными данными, информационной безопасностью и сопровождением программных сервисов. Практическая часть организована как сквозной профессиональный кейс, в котором обучающийся последовательно уточняет небольшую задачу транспортного программного сервиса, обосновывает отечественную и свободно распространяемую инструментальную среду, планирует работу, создает проверяемое изменение на Python, ведет историю Git, объединяет параллельные изменения, участвует во взаимном рецензировании, подтверждает результат автоматизированными проверками и разбирает обращение высокого приоритета без риска повторной обработки данных. Итогом становится связное портфолио в командном учебном репозитории, позволяющее проверить индивидуальный вклад студента, качество инженерных решений и готовность к профессиональной коммуникации.

Целью освоения дисциплины является формирование начальной профессиональной готовности обучающихся к обоснованному применению современных, отечественных и свободно распространяемых информационных технологий в воспроизводимом цикле командной разработки и сопровождения небольших программных изменений на Python.

Для достижения поставленной цели в рамках дисциплины решается комплекс задач, направленных на формирование у обучающихся способности – уточнять технологические и профессиональные требования к небольшой задаче, обосновывать выбор программно-аппаратной среды, декомпозировать и оценивать работы итерации, разрабатывать и проверять программное изменение на Python, вести содержательную историю Git, сравнивать и объединять параллельные версии кода, готовить изменение к независимому рецензированию, интерпретировать результаты автоматизированных проверок, анализировать журналы и данные при сопровождении транспортного сервиса, оформлять техническую

документацию и вести уважительную профессиональную коммуникацию с учетом различий ролей и культурных норм.

2. Планируемые результаты обучения по дисциплине (модулю).

Перечень формируемых результатов освоения образовательной программы (компетенций) в результате обучения по дисциплине (модулю):

ОПК-3 - Способен использовать современные информационные технологии и программно-аппаратные средства, в том числе отечественного производства, при решении задач профессиональной деятельности;

ППК-Р1 - Способен разрабатывать и отлаживать программный код;

ППК-Р6 - Способен участвовать в промышленной разработке программного обеспечения.

Обучение по дисциплине (модулю) предполагает, что по его результатам обучающийся будет:

Знать:

- место информатики и программной инженерии в системе профессиональной деятельности, роли участников команды и виды результатов разработки
- принципы представления информации, абстрагирования и моделирования программных систем
- архитектуру вычислительных систем, назначение процессора, памяти, устройств хранения и сетевых средств
- классы современных информационных технологий и программно-аппаратных средств, критерии их выбора, совместимости и совокупной стоимости владения
- назначение, архитектуру и базовые механизмы операционных систем, файловых систем, процессов, прав доступа и командной оболочки
- отечественные операционные системы, системы управления базами данных, платформы коллективной разработки и облачные решения, включенные в Единый реестр российского программного обеспечения
- сетевые модели, адресацию, протоколы прикладного уровня, программные интерфейсы и принципы клиент-серверной архитектуры
- жизненный цикл программного обеспечения, роли, рабочие продукты, контрольные точки и критерии готовности результата
- принципы инженерии требований, способы уточнения задачи, критерии приемки, ограничения и управление изменениями

- принципы гибкой разработки, итерационного планирования, оценки объема работ, управления потоком задач и командной ответственности

- понятие алгоритма, базовые алгоритмические конструкции, структуры данных и основы оценки вычислительной сложности

- синтаксическую и модульную организацию программ на Python, контракты функций, обработку исключений и журналирование

- принципы качества программного кода, правила именования, декомпозицию, рефакторинг и техническое документирование

- модель данных Git, назначение рабочего дерева, индекса, фиксаций, ветвей, меток и удаленных репозиториев

- регламенты ветвления, фиксации, сравнения, слияния и разрешения конфликтов при параллельной разработке

- процессы взаимного рецензирования кода, коллективного владения кодом и принятия изменений в защищенную ветвь

- уровни и виды тестирования, методы построения тестов, структуру тестового набора и назначение регрессионной проверки

- принципы управления зависимостями, воспроизводимого окружения, автоматизированной сборки и непрерывной интеграции

- реляционную модель данных, ключи, связи, ограничения целостности, транзакции и основы запросов SQL в PostgreSQL

- принципы наблюдаемости программных систем, структуру журналов, показатели состояния и порядок анализа инцидентов

- основные требования информационной безопасности при разработке, включая управление секретами, защиту данных, проверку зависимостей и безопасное изменение критичных функций

- нормы профессиональной коммуникации, делового этикета, межкультурного взаимодействия, инженерной этики и проверяемого применения систем искусственного интеллекта

Уметь:

- уточнять содержание небольшой задачи разработки с использованием шаблона карточки задачи и критериев приемки при неполных или противоречивых исходных требованиях

- декомпозировать и оценивать объем работы с использованием доски задач и метода относительной оценки при фиксированной продолжительности итерации, зависимостях и ограниченной доступности участников

- обоснованно выбирать информационные технологии и программно-аппаратные средства с использованием Единого реестра российского программного обеспечения, официальной документации и матрицы

критериев при ограничениях совместимости, безопасности и совокупной стоимости владения

- описывать контекст программного сервиса и маршрут обработки обращения с использованием контекстных схем, журналов и ролевой модели при наличии нескольких компонентов, участников и контрольных точек

- разрабатывать минимальное изменение программного модуля с использованием Python, стандартной библиотеки и правил PEP 8 при сохранении существующего контракта и установленной границы задачи

- разрабатывать автоматизированные проверки с использованием pytest и методов построения тестов при заданных критериях приемки и риске регрессионных нарушений

- регистрировать и сохранять изменения исходного кода с использованием Git и GitFlic при соблюдении регламента ветвления, содержательных фиксаций и исключения секретов из репозитория

- сравнивать и объединять версии исходного кода с использованием Git при параллельных изменениях и необходимости сохранить требуемое поведение всех ветвей

- подготавливать изменение к взаимному рецензированию с использованием запроса на включение изменений и шаблона проверки в GitFlic при запрете самостоятельного утверждения и наличии обязательных доказательств

- настраивать воспроизводимую автоматизированную проверку изменения с использованием GitFlic CI/CD, pytest и Ruff при изолированном окружении и обязательном успешном завершении конвейера

- анализировать обращение высокого приоритета с использованием карточки инцидента, журналов приложения и запросов PostgreSQL при ограниченном времени восстановления и риске повторной обработки данных

- представлять и аргументировать инженерный результат с использованием технической документации, карточки задачи и материалов репозитория при различиях профессиональных ролей, культурных норм и уровня технической подготовки адресатов

Владеть:

- навыками постановки и планирования небольшой задачи разработки с фиксацией критериев приемки, ограничений и доказательств готовности результата

- навыками разработки и отладки небольшого программного изменения на Python с применением pytest и Ruff в рабочем окружении

- навыками работы с Git в учебном репозитории GitFlic – создания ветви, выполнения содержательных фиксаций, сравнения и слияния изменений

- навыками подготовки изменения к командной проверке в GitFlic – оформления запроса на включение изменений, проведения взаимного рецензирования и аргументированной обработки замечаний

3. Объем дисциплины (модуля).

3.1. Общая трудоемкость дисциплины (модуля).

Общая трудоемкость дисциплины (модуля) составляет 5 з.е. (180 академических часа(ов).

3.2. Объем дисциплины (модуля) в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении учебных занятий:

Тип учебных занятий	Количество часов	
	Всего	Семестр №1
Контактная работа при проведении учебных занятий (всего):	64	64
В том числе:		
Занятия лекционного типа	32	32
Занятия семинарского типа	32	32

3.3. Объем дисциплины (модуля) в форме самостоятельной работы обучающихся, а также в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении промежуточной аттестации составляет 116 академических часа (ов).

3.4. При обучении по индивидуальному учебному плану, в том числе при ускоренном обучении, объем дисциплины (модуля) может быть реализован полностью в форме самостоятельной работы обучающихся, а также в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении промежуточной аттестации.

4. Содержание дисциплины (модуля).

4.1. Занятия лекционного типа.

№ п/п	Тематика лекционных занятий / краткое содержание
1	Информатика и профессиональная деятельность инженера-программиста Рассматриваемые вопросы: - предмет и методы информатики, связь с программной инженерией; - информация, данные, алгоритм и программная система как базовые понятия; - представление числовых, текстовых, графических и структурированных данных; - абстрагирование, моделирование и декомпозиция инженерных задач; - профессиональные роли, зоны ответственности и результаты командной разработки.
2	Архитектура вычислительных систем и программно-аппаратная среда Рассматриваемые вопросы: - принципы организации вычислительной системы и взаимодействие ее компонентов; - процессор, память, накопители и иерархия хранения данных; - периферийные и сетевые устройства, виртуализация вычислительных ресурсов; - классы программно-аппаратных средств и критерии выбора под профессиональную задачу.
3	Операционные системы и отечественная программная инфраструктура Рассматриваемые вопросы: - назначение и архитектура операционных систем, процессы и управление ресурсами; - файловые системы, права доступа, командная оболочка и переменные окружения; - семейство Linux и особенности рабочей среды разработчика; - отечественные операционные системы Astra Linux, ALT Linux, РЕД ОС и ОС «Аврора» в контексте технологического суверенитета; - назначение Единого реестра российского программного обеспечения и критерии совместимости.
4	Сетевое взаимодействие и архитектура программных сервисов Рассматриваемые вопросы: - модели OSI и TCP/IP, адресация и маршрутизация данных; - протоколы прикладного уровня, структура HTTP-запроса и ответа; - клиент-серверное взаимодействие и программные интерфейсы; - контекст программного сервиса, потоки данных и внешние зависимости; - модели локальной, облачной и гибридной инфраструктуры.
5	Жизненный цикл программного обеспечения и промышленный рабочий процесс Рассматриваемые вопросы: - стадии жизненного цикла программного обеспечения и их результаты; - роли заказчика, владельца продукта, аналитика, разработчика, специалиста по тестированию и сопровождению; - рабочие продукты команды, контрольные точки и критерии готовности; - итерационная и последовательная организация разработки; - прослеживаемость требований, изменений, проверок и выпусков.
6	Инженерия требований и постановка задачи разработки Рассматриваемые вопросы: - виды требований и свойства качественной формулировки; - наблюдаемый и ожидаемый результаты, ограничения и допущения; - критерии приемки, определение готовности и граница изменения; - методы уточнения неполной или противоречивой информации; - связь требований с проверками и доказательствами результата.
7	Планирование итерации и управление потоком работ Рассматриваемые вопросы: - декомпозиция задачи на проверяемые работы; - методы оценки объема, неопределенности и резерва; - зависимости, ограничения доступности и распределение ответственности; - состояния карточки задачи и правила актуализации доски; - принципы гибкой разработки и ежедневной синхронизации команды.

№ п/п	Тематика лекционных занятий / краткое содержание
8	Алгоритмизация и язык Python в инженерной разработке Рассматриваемые вопросы: - свойства алгоритма, способы описания и базовые управляющие конструкции; - типы данных, выражения, условные операторы и циклы Python; - коллекции, итерации и обработка текстовых и числовых данных; - функции, параметры, возвращаемые значения и область видимости; - оценка корректности, понятности и вычислительной эффективности алгоритма.
9	Модульная организация программ и обработка отказов Рассматриваемые вопросы: - модули, пакеты, зависимости и границы ответственности; - программные контракты, предусловия, постусловия и инварианты; - обработка исключений и формирование диагностических сообщений; - журналирование событий, уровни записей и идентификаторы трассировки; - минимальное изменение как способ снижения технического риска.
10	Модель данных Git и регистрация изменений Рассматриваемые вопросы: - назначение распределенной системы управления версиями; - рабочее дерево, индекс, фиксация и граф истории; - локальный и удаленный репозитории, ссылки и метки; - содержательная фиксация и связь изменения с задачей; - исключение временных файлов, результатов сборки и секретов.
11	Проверка и тестирование программного обеспечения Рассматриваемые вопросы: - цели, уровни и виды тестирования программного обеспечения; - методы построения тестов на основе классов эквивалентности, граничных значений и таблиц решений; - структура автоматизированной проверки и организация набора pytest; - положительные, отрицательные и регрессионные сценарии; - локализация дефекта, воспроизводимость и оформление результата проверки.
12	Ветвление, объединение и взаимное рецензирование кода Рассматриваемые вопросы: - назначение ветвей и стратегии их именования; - сравнение версий и анализ состава изменения; - способы объединения, причины конфликтов и порядок их разрешения; - запрос на включение изменений, роли автора и рецензента; - коллективное владение кодом и правила защищенной ветви.
13	Качество программного кода и техническая документация Рассматриваемые вопросы: - критерии читаемости, простоты, связности и сопровождаемости кода; - правила именования и оформления Python-кода по PEP 8; - декомпозиция, устранение дублирования и безопасный рефакторинг; - комментарии, строки документирования и описание программного интерфейса; - стандарты, шаблоны и прослеживаемость технической документации.
14	Управление зависимостями и непрерывная интеграция Рассматриваемые вопросы: - воспроизводимое окружение Python и управление зависимостями; - структура pyproject.toml и назначение инструментов качества; - автоматизированная сборка, проверка стиля и запуск тестов; - этапы, задания, исполнители и артефакты конвейера CI/CD; - условия допуска изменения к объединению и диагностика неуспешного запуска.

№ п/п	Тематика лекционных занятий / краткое содержание
15	Реляционные данные в программных системах Рассматриваемые вопросы: - сущности, атрибуты, ключи и связи реляционной модели; - ограничения целостности, нормализация и транзакции; - выборка, фильтрация, сортировка, группировка и соединение данных средствами SQL; - отечественные и открытые системы PostgreSQL и Postgres Pro в составе прикладной инфраструктуры; - безопасный доступ приложения к данным и миграция схемы.
16	Сопровождение, безопасность и ответственное развитие программных систем Рассматриваемые вопросы: - наблюдаемость сервиса, показатели состояния, журналы и диагностические сведения; - различия инцидента, запроса на обслуживание, проблемы и изменения; - приоритет, эскалация, безопасное восстановление и критерии закрытия обращения; - защита секретов, персональных данных, зависимостей и цепочки поставки программного обеспечения; - профессиональная этика, межкультурная коммуникация и проверяемое применение систем искусственного интеллекта.

4.2. Занятия семинарского типа.

Практические занятия

№ п/п	Тематика практических занятий/краткое содержание
1	Определение правил профессионального взаимодействия команды Студент сопоставляет роли участников учебной команды, различия коммуникативных норм и требования делового этикета. В текстовом процессоре платформы Р7 или в МойОфис он оформляет единый регламент рабочих сообщений, обсуждения замечаний и подтверждения договоренностей. Согласованный регламент применяется при ведении всех последующих материалов.
2	Обоснование инструментальной среды разработки Студент сравнивает Astra Linux, ALT Linux и РЕД ОС, платформы GitFlic и GitVerse, PostgreSQL и Postgres Pro по совместимости, условиям применения, безопасности и совокупной стоимости владения. В табличном процессоре он составляет матрицу критериев и выбирает основной набор средств для учебного репозитория. Принятое решение фиксируется в технической записке и задает ограничения для дальнейшей работы.
3	Контекстная модель транспортного программного сервиса Студент изучает состав сервиса на Python фоновых заданий, хранилища PostgreSQL, пользовательского интерфейса и внешних участников. Он строит контекстную схему, отмечает потоки данных, границы ответственности, точки регистрации событий и возможные отказы. Утвержденная схема используется для постановки задачи и последующего анализа обращения.
4	Уточнение программного изменения и критериев приемки Студент преобразует исходное сообщение о неверной проверке периода задания в карточку задачи с наблюдаемым и ожидаемым результатами. Он формулирует проверяемые критерии приемки, границу изменения, ограничения совместимости и определение готовности. Карточка становится основанием для оценки работ и разработки проверок.
5	Декомпозиция и оценка работ итерации Студент разделяет задачу на последовательные работы, определяет зависимости, оценивает объем и закладывает резерв на объединение параллельного изменения. В системе управления задачами он составляет трехдневный план с ответственными, контрольными точками и фактическими состояниями. План используется для организации истории изменений и контроля завершения.

№ п/п	Тематика практических занятий/краткое содержание
6	Регламент ветвления и содержательной фиксации изменений Студент определяет имя рабочей ветви, последовательность фиксаций, правила синхронизации с основной ветвью и состав исключаемых файлов. В локальном репозитории Git он создает ветвь от установленной исходной точки и проверяет связь с удаленным репозиторием GitFlic. Подготовленная структура истории задает порядок внесения проверок и программного изменения.
7	Разработка спецификации автоматизированных проверок Студент связывает каждый критерий приемки с положительным, отрицательным и граничным сценарием. Он подготавливает структуру тестовых данных и ожидаемых результатов для pytest, включая проверку независимости от часового пояса и сохранения существующего поведения. Спецификация определяет состав первой фиксации в рабочей ветви.
8	Реализация автоматизированных проверок на Python Студент создает проверки pytest для корректного, равного и неверного периода запуска и выполняет их на исходной версии модуля. Он фиксирует воспроизводимый неуспешный результат, затем регистрирует тесты отдельной содержательной фиксацией Git. Полученные доказательства используются для минимального исправления.
9	Внесение минимального изменения в программный модуль Студент изменяет модуль на Python проверки фонового задания в пределах утвержденной карточки задачи. Он обрабатывает неверный порядок временных значений, сохраняет существующие ошибки и добивается успешного прохождения всего набора pytest. Исправление регистрируется отдельной фиксацией и сопоставляется с критериями приемки.
10	Сравнение версий и объединение параллельных изменений Студент получает ветвь другого разработчика с добавлением приоритета задания и анализирует пересекающиеся изменения. Он выполняет объединение в Git, разрешает конфликт с сохранением приоритета и обеих проверок периода, затем повторно запускает проверки. Итоговая история и сравнение с основной ветвью становятся доказательствами целостности результата.
11	Подготовка запроса на включение изменений Студент размещает рабочую ветвь в GitFlic и оформляет запрос на включение изменений по принятому шаблону. Он связывает запрос с карточкой задачи, описывает состав изменения, способ проверки, риск и границу работы, прикладывает результаты тестов и граф истории. Материалы передаются другому студенту для независимого рецензирования.
12	Взаимное рецензирование программного кода Студент проверяет запрос другого участника по критериям корректности, читаемости, достаточности тестов и отсутствия несвязанных изменений. Он формулирует предметные замечания, а затем обрабатывает комментарии к собственной работе, фиксируя исправление, выделение отдельной задачи или технически обоснованное отклонение. Согласованная версия допускается к автоматизированной проверке.
13	Настройка воспроизводимой проверки изменения Студент подготавливает конфигурацию GitFlic CI для установки зависимостей, проверки оформления кода средствами Ruff и запуска pytest. Он запускает конвейер, анализирует журнал выполнения, устраняет причины неуспешного задания и сохраняет подтверждение успешного результата. Проверенный запрос становится исходным состоянием для разбора обращения высокого приоритета.
14	Классификация обращения высокого приоритета Студент анализирует неполное сообщение о частичном результате фоновой обработки, журнал приложения и результаты диагностических запросов к PostgreSQL по пяти пропущенным записям. Он оформляет карточку обращения с фактическим и ожидаемым результатами, масштабом, ограничениями, вопросами к оператору и наблюдаемыми критериями закрытия. Подтвержденная карточка служит основанием для выбора безопасных действий.
15	Маршрут обработки обращения и безопасный план действий Студент строит последовательность передачи обращения между оператором, линиями

№ п/п	Тематика практических занятий/краткое содержание
	сопровождения, разработчиком, владельцем продукта и ответственным за выпуск. Он распределяет действия до контрольного срока, оценивает риск повторной обработки, исключает создание дубликатов и определяет доказательство завершения каждого перехода. Схема маршрута и план включаются в итоговый комплект материалов.
16	Представление результата инженерной итерации Студент актуализирует доску задач, собирает карточки, технические решения, историю Git, результаты рецензирования, журнал конвейера и материалы по обращению в единое портфолио. Он готовит отдельные сообщения для пользователя и команды, отделяя подтвержденные факты от предположений и адаптируя уровень подробности к адресату. На индивидуальной защите студент демонстрирует свой вклад, обосновывает решения и отвечает на вопросы по оставшимся рискам.

4.3. Самостоятельная работа обучающихся.

№ п/п	Вид самостоятельной работы
1	Изучение рекомендованной литературы.
2	Подготовка к практическим работам.
3	Подготовка к промежуточной аттестации.
4	Подготовка к текущему контролю.

5. Перечень изданий, которые рекомендуется использовать при освоении дисциплины (модуля).

№ п/п	Библиографическое описание	Место доступа
1	Тюленева, Т. А. Информационные технологии в профессиональной деятельности : учебное пособие / Т. А. Тюленева. – Кемерово : КузГТУ имени Т. Ф. Горбачева, 2023. – 236 с. – ISBN 978-5-00137-449-7. – Текст : электронный	Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/399785 (дата обращения: 27.08.2026).
2	Турнецкая, Е. Л. Программная инженерия. Интеграционный подход к разработке : учебник для вузов / Е. Л. Турнецкая, А. В. Аграновский. – Санкт-Петербург : Лань, 2023. – 216 с. – ISBN 978-5-507-46898-0. – Текст : электронный	Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/352307 (дата обращения: 27.08.2026).
3	Васильева, М. А. Система контроля версий. Основы командной разработки : учебное пособие для вузов / М. А. Васильева, К. М. Филипченко. – Санкт-Петербург : Лань, 2022. – 144 с. – ISBN 978-5-507-44630-8. – Текст : электронный	Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/261089 (дата обращения: 27.08.2026).
4	Кочетыгов, А. А. Основы программирования на языке Python : учебное пособие / А. А. Кочетыгов. – Тула : ТулГУ, 2024. – 272 с. – ISBN 978-5-7679-5380-6. – Текст : электронный	Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/427316 (дата обращения: 27.08.2026).

5	Игнатьев, А. В. Тестирование программного обеспечения : учебное пособие для вузов / А. В. Игнатьев. – 4-е изд., стер. – Санкт-Петербург : Лань, 2025. – 56 с. – ISBN 978-5-507-50858-7. – Текст : электронный	Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/481331 (дата обращения: 27.08.2026).
6	Волк, В. К. Базы данных : учебник / В. К. Волк, В. Ю. Осеев, О. С. Черепанов. – Вологда : Инфра-Инженерия, 2025. – 544 с. – ISBN 978-5-9729-2594-0. – Текст : электронный	Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/499772 (дата обращения: 27.08.2026).
7	Воробьев, С. П. Компьютерные сети и сетевая безопасность : учебное пособие / С. П. Воробьев, С. Н. Широбокова, Р. К. Литвяк. – Новочеркасск : ЮРГПУ (НПИ), 2022. – 216 с. – ISBN 978-5-9997-0805-2. – Текст : электронный	Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/292247 (дата обращения: 27.08.2026).

6. Перечень современных профессиональных баз данных и информационных справочных систем, которые могут использоваться при освоении дисциплины (модуля).

- Электронно-библиотечная система «Лань». – URL: https://e.lanbook.com/ (дата обращения: 27.08.2026).

- Реестры российского и евразийского программного обеспечения Минцифры России. – URL: https://digital.gov.ru/activity/gos-uslugi/reestry-programmnogo-obespecheniya (дата обращения: 27.08.2026).

- Официальная документация Python 3. – URL: https://docs.python.org/3/ (дата обращения: 27.08.2026).

- Официальная книга и справочная документация Git. – URL: https://git-scm.com/book/ru/v2 (дата обращения: 27.08.2026).

- Документация GitFlic, включая управление запросами на включение изменений и CI/CD. – URL: https://docs.gitflic.ru/ (дата обращения: 27.08.2026).

- Официальная документация pytest. – URL: https://docs.pytest.org/ (дата обращения: 27.08.2026).

- Официальная документация Ruff. – URL: https://docs.astral.sh/ruff/ (дата обращения: 27.08.2026).

- Документация для разработчиков Astra Linux. – URL: https://docs.astralinux.ru/ (дата обращения: 27.08.2026).

- Документация PostgreSQL на русском языке от Postgres Professional. – URL: https://postgrespro.ru/docs/postgresql (дата обращения: 27.08.2026).

- Портал разработчиков ОС «Аврора». – URL: https://developer.auroraos.ru/ (дата обращения: 27.08.2026).

7. Перечень лицензионного и свободно распространяемого программного обеспечения, в том числе отечественного производства, необходимого для освоения дисциплины (модуля).

- Операционная система: Astra Linux Special Edition; допустимые учебные аналоги – ALT Linux или РЕД ОС.
- Подготовка технической документации: Р7-Офис или МойОфис.
- Разработка на Python: Python 3.11+, venv, pip, VSCodium или PyCharm Community.
- Управление версиями и командная работа: Git, GitFlic; допустимый отечественный аналог – GitVerse.
- Проверка качества и автоматизация: pytest, Ruff, GitFlic CI.
- Работа с данными: PostgreSQL или Postgres Pro, DBeaver Community или pgAdmin

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

Учебные аудитории для проведения учебных занятий, оснащенные компьютерной техникой и наборами демонстрационного оборудования.

Для практических занятий – наличие персональных компьютеров вычислительного класса.

9. Форма промежуточной аттестации:

Экзамен в 1 семестре.

10. Оценочные материалы.

Оценочные материалы, применяемые при проведении промежуточной аттестации, разрабатываются в соответствии с локальным нормативным актом РУТ (МИИТ).

Авторы:

старший преподаватель кафедры
«Цифровые технологии управления
транспортными процессами»
Института железнодорожного
транспорта А.Ю. Кремнев

Согласовано:

Заведующий кафедрой ЦТУТП В.Е. Нутович

Директор ЦС IT - программ ВИШ О.Б. Проневич

Председатель учебно-методической
комиссии Н.А. Андриянова