МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ ТАДЖИКИСТАН
МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕВЫСШЕГО ОБРАЗОВАНИЯ «РОССИЙСКО-ТАДЖИКСКИЙ (СЛАВЯНСКИЙ) УНИВЕРСИТЕТ»

«Утверждаю»
Декан Естественнонаучного факультета
_____
«29» 08 2025г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

ПРОГРАММНАЯ ИНЖЕНЕРИЯ

Направление подготовки – 09.03.03 «Прикладная информатика»
Профиль подготовки «Прикладная информатика инженерия программного обеспечения»
Форма подготовки - очная
Уровень подготовки – бакалавриат

ДУШАНБЕ 2025

Рабочая программа составлена в соответствии с требованиями федерального государственного образовательного стандарта высшего образования, утвержденного приказом Минобрнауки РФ от 19 сентября 2017г. № 922

При разработке рабочей программы учитываются:

- требования работодателей, профессиональных стандартов по направлению;
- содержание программ дисциплин, изучаемых на предыдущих и последующих этапах обучения;
- новейшие достижения в данной предметной области.

Рабочая программа обсуждена на заседании кафедры Информатики и ИТ, протокол №1 «18» августа 2025.

Рабочая программа утверждена УМС естественнонаучного факультета, протокол №1 «25» августа 2025.

Рабочая программа утверждена Учёным советом естественнонаучного факультета, протокол №1 «28» августа 2025.

Заведующий кафедрой, к.э.н., доцент: / _____ / Лешукович А. И.

Зам. председателя УМС факультета
ст. преподаватель: / _____ /Мирзокаримов О.А.

Разработчик, ст. преподаватель: / _____ / Мирзокаримов О.А.

Разработчик от организации: / _____ / Саидов И.Дж.
Менеджер по внедрению систем автоматизации в
ООО «Авесто групп» / Avesto Group LLC.

Расписание занятий дисциплины

Ф.И.О. преподавателя	Аудиторные занятия				Место работы преподавателя
	Лекция	**Практические занятия**	**(КСР, лаб.)**	**Приём СРС**	РТСУ, кафедра информатики и ИТ, Корпус 2, 216 каб.
Мирзокаримов О.А.	**Вторник**	**Четверг**	**Пятница**	**Пятница**	
	08:00-09:20	09:30-10:50	11:00-12:40	**14:10 15:30**	
	второй корпус:	второй корпус:	второй корпус:		
	Ауд.213	Ауд.221	Ауд.223	Каб. 210	

1. ЦЕЛИ И ЗАДАЧИ ИЗУЧЕНИЯ ДИСЦИПЛИНЫ

1.1. Цели изучения дисциплины

Целями освоения дисциплины "Программная инженерия" является приобретение комплекса теоретических знаний и методологических основ в области разработки программного обеспечения, а также практических навыков, необходимых для квалифицированного управления процессами создания программных продуктов на всех этапах жизненного цикла. Дисциплина является важной составной частью подготовки специалиста в области информационных технологий и разработки ПО. Основой курса являются современные методологии программной инженерии и процессные модели разработки, позволяющие при создании программных систем решить следующие основные задачи: - обеспечение качества программного продукта через применение инженерных подходов к разработке; - эффективное управление требованиями и их трассировка на протяжении всего жизненного цикла; - проектирование архитектуры программных систем с учетом принципов модульности, расширяемости и сопровождаемости; - внедрение процессов верификации, валидации и управления конфигурацией. Программой курса предусматривается изучение современных инструментов программной инженерии, включая системы управления версиями, средства автоматизации тестирования, инструменты непрерывной интеграции и развертывания.

1.2. Задачи изучения дисциплины

Задачи дисциплины формулируются в соответствии с требованиями ФГОС, предъявляемые к компетенциям обучающегося в области разработки и сопровождения программного обеспечения.

1.3. В результате изучения дисциплины «Программная инженерия» у обучающихся формируются следующие общепрофессиональные компетенции:

Таблица 1.

Код компетенции	Результаты освоения ОПОП	Перечень планируемых результатов обучения	Вид оценочного знания
ПК-1	**ПК-1**. Способен проводить обследование организаций, выявлять информационные потребности пользователей, формировать требования к информационной системе.	**ИПК-1.1.** Использует методику проведения обследования организации и выявления информационных потребностей пользователей **ИПК-1.2.** Анализирует деятельности предприятий, и выявляет участки производства, нуждающиеся в автоматизации **ИПК-1.3.** Осуществляет широкой общей подготовкой (базовыми знаниями) для решения практических задач в области информационных систем и технологий; теоретическими знаниями о роли компьютерных систем управления информационными потоками; типовыми разработанными средствами защиты информации и возможностями их использования в реальных задачах создания и внедрения информационных систем; навыками выбора класса ИС для автоматизации предприятия в соответствии с требованиями к ИС и ограничениями; способами автоматизации для конкретного предприятия; способами выбора ИС на основании преимуществ и недостатков существующих способов; расчета совокупной стоимостизвладения ИС; способами	Отчеты по практическим работам. Устный опрос. Презентация

		организации стратегического и оперативного планирования ИС.	
ОПК-4	**ОПК-4.** Способен участвовать в разработке стандартов, норм и правил, а также технической документации, связанной с профессиональной деятельностью	**ИОПК-4.1**. Знает основные стандарты оформления технической документации на различных стадиях жизненного цикла информационной системы. **ИОПК-4.2.** Применяет стандарты, нормы и правила оформления технической документации на различных стадиях жизненного цикла информационной системы. ИОПК-4.3. Разрабатывает *техническую* документацию на различных этапах жизненного цикла информационной системы.	Отчеты по практическим работам. Устный опрос. Презентация
ОПК-7	**ОПК-7.** Способен разрабатывать алгоритмы и программы, пригодные для практического применения	**ИОПК-7.1.** Применяет языки программирования и работы с базами данных, операционные системы и оболочки, современные программные среды разработки информационных систем и технологий. **ИОПК-7.2.** Применяет языки программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ. **ИОПК-7.3.** Программирует, выполняет отладку и тестирование прототипов программно-технических комплексов задач.	Отчеты по практическим работам. Устный опрос. Презентация
ОПК-8	Способен принимать участие в управлении проектами создания информационных систем на стадиях жизненного цикла	**ИОПК-8.1.** Применяет основные технологии создания и внедрения информационных систем, стандарты управления жизненным циклом информационной системы. **ИОПК-8.2.** Осуществляет организационное обеспечение выполнения работ на всех стадиях и в процессах жизненного цикла информационной системы. **ИОПК-8.3.** Составляет плановую и отчетную документацию по управлению проектами создания информационных систем на стадиях жизненного цикла.	Отчеты по практическим работам. Устный опрос. Презентация

2.МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ООП

Данная дисциплина входит в базовый цикл вариативной части дисциплины Б1.0.27 ОПОП бакалавриата ФГОС ВО и является обязательной дисциплиной.

Таблица 2.

№ п/п	Наименование дисциплины	Семестр	Место дисциплины в структуре ООП
1.	*Программирование*	1-2-3	Б1.О.13
2.	*Практикум по программированию*	2-4	Б1.О.21
3.	*Интеллектуальные информационные системы*	5	Б1.В.05
4.	*Проектирование информационных систем*	5-6	Б1.О.26
5.	*Информационные системы и технологии*	5	Б1.О.25
6.	*Теория систем и системный анализ*	7	Б1.В.12
7.	*Ознакомительная практика*	4	Б2.О.01
8.	*Технологическая (проектно-технологическая) практика*	6	Б2.О.02
9.	*Разработка системы электронного документооборота*	*7*	Б1.В.10
10.	*Преддипломная практика*	8	Б2.В.01

При освоении данной дисциплины необходимы умения и готовность («входные» знания») обучающегося по дисциплинам 1-2, указанных в Таблице 2. Дисциплины 3-10 относятся к группе, которые должны использовать «входные» знания данной дисциплины.

3. СТРУКТУРА И СОДЕРЖАНИЕ КУРСА, КРИТЕРИИ НАЧИСЛЕНИЯ БАЛЛОВ

Объем дисциплины *составляет 4 зачетных единиц, всего 144 часов, из которых: лекции 16 часов, практические занятия 16 часа, лабораторные работы 16 часов, КСР – 16 часа, всего часов аудиторной нагрузки - 64 часа, самостоятельная работа – 26 часов, контроль – 54 часов.*
экзамен –5-й семестр

3.1 Структура и содержание теоретической части курса

5-й семестр

Лекция 1. Введение в программную инженерию и ее дисциплины (2 часа)

Программная инженерия представляет междисциплинарную область знаний, объединяющую научные принципы и инженерные методы для создания качественных программных систем. Становление программной инженерии как отдельной дисциплины обусловлено необходимостью систематического подхода к разработке сложных программных продуктов. Пять основных дисциплин программной инженерии - научная, инженерная, производственная, управленческая и экономическая - формируют комплексный подход к созданию программного обеспечения. Кризис программного обеспечения 1960-х годов продемонстрировал важность применения инженерных принципов к разработке ПО. Этические аспекты программной инженерии определяют профессиональную ответственность разработчиков перед обществом и заказчиками.

Лекция 2. Области знаний SWEBOK и жизненный цикл ПО (2 часа)

Стандарт SWEBOK определяет структуру знаний в программной инженерии через пять основных областей: инженерия требований, проектирование, конструирование, тестирование и сопровождение. Жизненный цикл программного обеспечения охватывает все этапы от концепции до завершения эксплуатации системы. Различные модели жизненного цикла - каскадная, итеративная, спиральная - предоставляют альтернативные подходы к организации процесса разработки. Выбор модели жизненного цикла определяется характеристиками проекта, требованиями заказчика и ограничениями ресурсов. Понимание жизненного цикла обеспечивает системный подход к планированию и управлению проектами разработки ПО.

Лекция 3. Инженерия требований и моделирование ПО (2 часа)

Инженерия требований представляет систематический процесс выявления, анализа, документирования и валидации потребностей заинтересованных сторон. Функциональные требования определяют поведение системы, а нефункциональные требования задают ограничения качества и производительности. Моделирование программных систем использует абстрактные представления для понимания сложности и взаимосвязей компонентов. Процесс сбора требований включает интервью, наблюдение, анализ документов и прототипирование. Трассировка требований обеспечивает связность между потребностями пользователей и реализацией системы. Управление изменениями требований критично для успеха проектов в условиях эволюционирующих потребностей.

Лекция 4. Архитектура программного обеспечения (2 часа)

Архитектура программного обеспечения определяет высокоуровневую структуру системы и взаимодействие между основными компонентами. Архитектурные решения влияют на качественные атрибуты системы: производительность, масштабируемость, надежность и сопровождаемость. Архитектурные стили и паттерны предоставляют проверенные решения для типовых проблем проектирования. Выбор архитектуры определяет технологический стек, распределение ответственности между компонентами и стратегию интеграции. Документирование архитектуры обеспечивает понимание системы всеми участниками проекта. Оценка архитектурных решений включает анализ рисков и соответствие требованиям качества.

Лекция 5. Шаблоны проектирования и парадигмы программирования (2 часа)

Шаблоны проектирования представляют повторно используемые решения типовых проблем объектно-ориентированного программирования. Классификация паттернов на порождающие, структурные и поведенческие обеспечивает систематический подход к их применению. Парадигмы программирования определяют концептуальные модели разработки: императивная, объектно-ориентированная, функциональная и логическая. Выбор парадигмы влияет на структуру программы, методы декомпозиции задач и стиль программирования. Метапрограммирование расширяет возможности языков программирования через генерацию и модификацию кода. Современные языки программирования часто поддерживают несколько парадигм для решения различных типов задач.

Лекция 6. Тестирование, верификация и валидация ПО (2 часа)

Тестирование представляет процесс верификации соответствия программы требованиям, включающий модульное, интеграционное и системное тестирование. Верификация отвечает на вопрос "Правильно ли разрабатывается продукт?", а валидация - "Правильный ли продукт разрабатывается?". Методы тестирования включают статический анализ кода, динамическое тестирование и формальную верификацию. Стратегии тестирования определяют последовательность и глубину проверок на различных уровнях системы. Автоматизация тестирования повышает эффективность процесса и

обеспечивает регрессионное тестирование. Планирование тестирования включает разработку тест-кейсов, критериев завершения и метрик качества.

Лекция 7. Эволюция ПО и управление изменениями (2 часа)

Эволюция программного обеспечения представляет непрерывный процесс адаптации системы к изменяющимся требованиям и условиям эксплуатации. Управление изменениями обеспечивает контролируемую модификацию программных артефактов на всех этапах жизненного цикла. Конфигурационное управление включает идентификацию, контроль статуса, аудит и отчетность изменений. Системы управления версиями фиксируют историю изменений и обеспечивают параллельную разработку. Стратегии ветвления определяют организацию разработки новых функций и исправления дефектов. Процесс релиза координирует интеграцию изменений и развертывание новых версий системы.

Лекция 8. Управление проектами и качество ПО (2 часа)

Управление программными проектами интегрирует планирование, координацию ресурсов и контроль выполнения для достижения целей проекта. Планирование включает декомпозицию работ, оценку трудозатрат, составление расписания и распределение ресурсов. Управление рисками идентифицирует потенциальные угрозы проекту и разрабатывает стратегии их минимизации. Качество программного обеспечения определяется соответствием функциональным требованиям и достижением нефункциональных характеристик. Процессы обеспечения качества включают планирование, контроль и улучшение качества на всех этапах разработки. Документирование ПО обеспечивает понимание системы разработчиками и пользователями, включая техническую документацию и руководства пользователя.

3.2. Структура и содержание практической части курса

Структура и содержание практической части курса включает в себя тематику и содержание практических занятий (ПЗ) и лабораторных работ.

5-й семестр

Практические занятия (16 часов)

Практическое занятие 1. Анализ дисциплин программной инженерии (2 часа)
Практическое занятие 2. Инженерия требований по методологии SWEBOK (2 часа)
Практическое занятие 3. Моделирование и проектирование архитектуры (2 часа)
Практическое занятие 4. Применение шаблонов проектирования (2 часа)
Практическое занятие 5. Планирование тестирования и валидации (2 часа)
Практическое занятие 6. Управление эволюцией ПО (2 часа)
Практическое занятие 7. Управление программными проектами (2 часа)
Практическое занятие 8. Обеспечение качества ПО (2 часа)

Лабораторные работы (16 часов)

Лабораторная работа 1. Инженерия требований (2 часа)
Лабораторная работа 2. UML-моделирование (2 часа)
Лабораторная работа 3. Проектирование архитектуры (2 часа)
Лабораторная работа 4. Разработка в команде (2 часа)
Лабораторная работа 5. Тестирование ПО (2 часа)
Лабораторная работа 6. Управление конфигурацией (2 часа)
Лабораторная работа 7. Рефакторинг и качество кода (2 часа)
Лабораторная работа 8. Развертывание и мониторинг (2 часа)

Структура и содержание КСР (16 часов)

КСР 1. Анализ case study (2 часа)
КСР 2. Исследование инструментов (2 часа)
КСР 3. Разработка технического задания (2 часа)
КСР 4. Проектирование архитектуры (2 часа)
КСР 5. Планирование проекта (2 часа)
КСР 6. Исследование методологий (2 часа)
КСР 7. Анализ качества ПО (2 часа)
КСР 8. Подготовка к итоговому контролю (2 часа).

Таблица 3.

№ п/п	Раздел дисциплины	Виды учебной работы, включая самостоятельную работу студентов и трудоемкость (в часах)					Литература	Кол-во баллов в неделю
	5 семестр	Лек.	Пр.	Лаб.	КСР	СРС		
1.	**Лекция 1. Введение в программную инженерию и ее дисциплины** -Программная инженерия представляет междисциплинарную область знаний, объединяющую научные принципы и	2				2	1(5-15) 5 (с.11-16),	12,5

	инженерные методы для создания качественных программных систем. Становление программной инженерии как отдельной дисциплины обусловлено необходимостью систематического подхода к разработке сложных программных продуктов. Пять основных дисциплин программной инженерии - научная, инженерная, производственная, управленческая и экономическая - формируют комплексный подход к созданию программного обеспечения. Кризис программного обеспечения 1960-х годов продемонстрировал важность применения инженерных принципов к разработке ПО. Этические аспекты программной инженерии определяют профессиональную ответственность разработчиков перед обществом и заказчиками.						11 (с.31-49) 16 (с.19-33)	
2.	*Практическое занятие 1. Анализ дисциплин программной инженерии*		2					
3.	*Лабораторная работа 1. Инженерия требований*			2				
4.	*КСР 1. Анализ case study*				2	2		
5.	**Лекция 2. Области знаний SWEBOK и жизненный цикл ПО** Стандарт SWEBOK определяет структуру знаний в программной инженерии через пять основных областей: инженерия требований, проектирование, конструирование, тестирование и сопровождение. Жизненный цикл программного обеспечения охватывает все этапы от концепции до завершения эксплуатации системы. Различные модели жизненного цикла - каскадная, итеративная, спиральная - предоставляют альтернативные подходы к организации процесса разработки. Выбор модели жизненного цикла определяется характеристиками проекта, требованиями заказчика и ограничениями ресурсов. Понимание жизненного цикла обеспечивает системный подход к планированию и управлению проектами разработки ПО.	2				2	4(с.46-51) 3(с.92-93) 4(с.51-58) 3(с.94-96)	12,5
6.	*Практическое занятие 2. Инженерия требований по методологии SWEBOK*		2					
7.	*Лабораторная работа 2. UML-*			2				
8.	*КСР 2. Исследование инструментов*				2			
9.	**Лекция 3. Инженерия требований и моделирование ПО** Инженерия требований представляет систематический процесс выявления, анализа, документирования и валидации потребностей заинтересованных сторон. Функциональные требования определяют поведение системы, а нефункциональные требования задают ограничения качества и производительности. Моделирование программных систем использует абстрактные представления для понимания сложности и взаимосвязей компонентов. Процесс сбора требований включает интервью, наблюдение, анализ документов и прототипирование. Трассировка требований обеспечивает связность между потребностями пользователей и реализацией системы. Управление изменениями требований критично для успеха проектов в условиях эволюционирующих потребностей.	2				2	1(с.32-36) 5(с.213-220) 11(с.97-120) 4(с.99-102) 1(с.36-37)	12,5

10.	*Практическое занятие 3. Моделирование и проектирование архитектуры*		2					
11.	*Лабораторная работа 3. Проектирование архитектуры*			2				
12.	*КСР 3. Разработка технического задания*				2	2		
13.	**Лекция 4. Архитектура программного обеспечения.** Архитектура программного обеспечения определяет высокоуровневую структуру системы и взаимодействие между основными компонентами. Архитектурные решения влияют на качественные атрибуты системы: производительность, масштабируемость, надежность и сопровождаемость. Архитектурные стили и паттерны предоставляют проверенные решения для типовых проблем проектирования. Выбор архитектуры определяет технологический стек, распределение ответственности между компонентами и стратегию интеграции. Документирование архитектуры обеспечивает понимание системы всеми участниками проекта. Оценка архитектурных решений включает анализ рисков и соответствие требованиям качества.	2				2	1(с.37-39) 5(с.37-91) 11(с.97-120) 5 (с.91-93) 1 (с.115-116)	12,5
14.	*Практическое занятие 4. Применение шаблонов проектирования*		2					
15.	*Лабораторная работа 4. Разработка в команде*			2				
16.	*КСР 4. Проектирование архитектуры*				2	2		
17.	**Лекция 5. Шаблоны проектирования и парадигмы программирования** Шаблоны проектирования представляют повторно используемые решения типовых проблем объектно-ориентированного программирования. Классификация паттернов на порождающие, структурные и поведенческие обеспечивает систематический подход к их применению. Парадигмы программирования определяют концептуальные модели разработки: императивная, объектно-ориентированная, функциональная и логическая. Выбор парадигмы влияет на структуру программы, методы декомпозиции задач и стиль программирования. Метапрограммирование расширяет возможности языков программирования через генерацию и модификацию кода. Современные языки программирования часто поддерживают несколько парадигм для решения различных типов задач.	2				2	1(с.50-59) 21(с.149-220) 5(с.162, 178, 221) 1(с.116-117)	12,5
18.	*КСР 5. Планирование проекта*		2					
19.	*Лабораторная работа 5. Тестирование ПО*			2				
20.	*КСР 5. Планирование проекта*				2	2		
21.	**Лекция 6. Тестирование, верификация и валидация ПО** Тестирование представляет процесс верификации соответствия программы требованиям, включающий модульное, интеграционное и системное тестирование. Верификация отвечает на вопрос "Правильно ли разрабатывается продукт?", а валидация - "Правильный ли продукт разрабатывается?". Методы тестирования включают статический анализ кода, динамическое тестирование и формальную верификацию. Стратегии тестирования определяют последовательность и глубину	2				2	1(с.39-49) 5(с.316-366) 11(с.135-154) 16(с.224-241) 5(с.371-374) 1(с.118-119)	12,5

	проверок на различных уровнях системы. Автоматизация тестирования повышает эффективность процесса и обеспечивает регрессионное тестирование. Планирование тестирования включает разработку тест-кейсов, критериев завершения и метрик качества.							
22.	*Практическое занятие 6. Управление эволюцией ПО*		2					
23.	*Лабораторная работа 6. Управление конфигурацией*			2				
24.	*КСР 6. Исследование методологий*				2			
25.	**Лекция 7. Эволюция ПО и управление изменениями** Эволюция программного обеспечения представляет непрерывный процесс адаптации системы к изменяющимся требованиям и условиям эксплуатации. Управление изменениями обеспечивает контролируемую модификацию программных артефактов на всех этапах жизненного цикла. Конфигурационное управление включает идентификацию, контроль статуса, аудит и отчетность изменений. Системы управления версиями фиксируют историю изменений и обеспечивают параллельную разработку. Стратегии ветвления определяют организацию разработки новых функций и исправления дефектов. Процесс релиза координирует интеграцию изменений и развертывание новых версий системы.	2				2	1(с.64-82) 8 (с.57-105) 17(с.34-38) 17(с.39-41) 2(с.10-11)	12,5
26.	*Практическое занятие 7. Управление программными проектами*		2					
27.	*Лабораторная работа 7. Рефакторинг и качество кода*			2				
28.	*КСР 7. Анализ качества ПО*				2			
29.	**Лекция 8. Управление проектами и качество ПО** Управление программными проектами интегрирует планирование, координацию ресурсов и контроль выполнения для достижения целей проекта. Планирование включает декомпозицию работ, оценку трудозатрат, составление расписания и распределение ресурсов. Управление рисками идентифицирует потенциальные угрозы проекту и разрабатывает стратегии их минимизации. Качество программного обеспечения определяется соответствием функциональным требованиям и достижением нефункциональных характеристик. Процессы обеспечения качества включают планирование, контроль и улучшение качества на всех этапах разработки. Документирование ПО обеспечивает понимание системы разработчиками и пользователями, включая техническую документацию и руководства пользователя.	2				2	1(с.83-96) 8 (с.106-125) 17(с.42-75) 13(с.47-56) 17(с.76-85) 2(с.12-13)	12,5
30.	*Практическое занятие 8. Обеспечение качества ПО*		2					
31.	*Лабораторная работа 8. Развертывание и мониторинг*			2				
32.	*КСР 8. Подготовка к итоговому контролю.*				2	2		
	ИТОГО:	**16**	**16**	**16**	**16**	**26**		**200**

Формы контроля и критерии начисления баллов

Контроль усвоения студентом каждой темы осуществляется в рамках балльно-рейтинговой системы (БРС), включающей текущий, рубежный и итоговый контроль. Студенты **3-го курса**, обучающиеся по кредитно-рейтинговой системе обучения, могут получить максимально возможное

количество баллов - 300. Из них на текущий и рубежный контроль выделяется 200 баллов или 49% от общего количества.

На итоговый контроль знаний студентов выделяется 51% или 100 баллов.

Порядок выставления баллов: 1-й рейтинг (1-7 недели до 12,5 баллов+12,5 баллов (8 неделя – Рубежный контроль №1) = 100 баллов), 2-й рейтинг (9-15 недели до 12,5 баллов+12,5 баллов (16 неделя – Рубежный контроль №2) = 100 баллов), итоговый контроль 100 баллов.

К примеру, за текущий и 1-й рубежный контроль выставляется 100 баллов: лекционные занятия – 21 балл, за практические занятия (КСР, лабораторные) – 31,5 балл, за СРС – 17,5 баллов, требования ВУЗа – 17,5 баллов, рубежный контроль – 12,5 баллов.

В случае пропуска студентом занятий по уважительной причине (при наличии подтверждающего документа) в период академической недели деканат факультета обращается к проректору по учебной работе с представлением об отработке студентом баллов за пропущенные дни по каждой отдельной дисциплине с последующим внесением их в электронный журнал.

Итоговая форма контроля по дисциплине (зачет, экзамен) проводится как в форме тестирования, так и в традиционной (устной) форме. Тестовая форма итогового контроля по дисциплине предусматривает: для естественнонаучных направлений – 10 тестовых вопросов на одного студента, где правильный ответ оценивается в 10 баллов, для гуманитарных направлений – 25 тестовых вопросов, где правильный ответ оценивается в 4 балла. Тестирование проводится в электронном виде, устный экзамен на бумажном носителе с выставлением оценки в ведомости по аналогичной системе с тестированием.

Таблица 4.

Неделя	Активное участие на лекционных занятиях, написание конспекта и выполнение других видов работ*	Активное участие на практических (семинарских) занятиях, КСР	СРС Написание реферата, доклада, эссе Выполнение других видов работ	Выполнение положения высшей школы (установленная форма одежды, наличие рабочей папки, а также других пунктов устава высшей школы)	РК №1	Всего
1	2	3	4	5	6	7
1	3	4,5	2,5	2,5	-	12,5
2	3	4,5	2,5	2,5	-	12,5
3	3	4,5	2,5	2,5	-	12,5
4	3	4,5	2,5	2,5	-	12,5
5	3	4,5	2,5	2,5	-	12,5
6	3	4,5	2,5	2,5	-	12,5
7	3	4,5	2,5	2,5	-	12,5
8	-	-	-	-	12,5	12,5
Первый рейтинг	21	31,5	17,5	17,5	12,5	100

Формула вычисления результатов дистанционного контроля и итоговой формы контроля по дисциплине за семестр для студентов 4-го курсов:

$$ИБ = \left[\frac{(P_1 + P_2)}{2}\right] \cdot 0{,}49 + Эи \cdot 0{,}51$$

где ИБ –итоговый балл, P_1- итоги первого рейтинга, P_2- итоги второго рейтинга, *Эи–* результаты итоговой формы контроля (экзамен).

4. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

Учебно-методическое обеспечение самостоятельной работы обучающихся по дисциплине «Методы обработки информации» включает в себя:

1. план-график выполнения самостоятельной работы по дисциплине, в том числе примерные нормы времени на выполнение по каждому заданию;

2. характеристика заданий для самостоятельной работы обучающихся и методические рекомендации по их выполнению;

3. требования к представлению и оформлению результатов самостоятельной работы;
4. критерии оценки выполнения самостоятельной работы.

План-график выполнения самостоятельной работы по дисциплине

4.1. План-график выполнения самостоятельной работы по дисциплине

№	Объем СРС	Тема СРС	Форма и вид результатов самостоятельной работы	Форма контроля
1.	2	Основные этапы разработки программного обеспечения. Построение модели.	Вопросы 1-4. Описание технологии разработки, реферат	Опрос
2.	2	Методы и технологии проектирования ИС. Классификация методов проектирования.	Вопросы 5-8. Презентация методов	Выступление
3.	2	Методы быстрой разработки ПО. Подход RAD	Вопросы 8-10. Презентация, доклад	Выступление
4.	2	Проектирование структуры программного обеспечения. Обработка списков	Вопросы 11-13. Выполнение задания 1 (1-10).	Защита работы. Выступление
5.	2	Проектирование информационного обеспечения ИС. Моделирование данных	Выполнение задания 1. Конспект, презентация (вопросы 14-15)	Опрос, Выступление
6.	2	Моделирование потоков данных.	Выполнение задания 2	Защита работы.
7.	2	Структурный подход.	Вопросы 16-17. Выполнение задания 3	Защита работы.
8.	2	Метод функционального моделирования – SADT	Вопросы 16-17. Выполнение задания 4	Защита работы.
9.	2	Структурное программирование	Выполнение задания 5	Защита работы.
10.	2	Основные понятия объектно-ориентированного подхода	Вопросы 18-25. Выполнение задания 6	Защита работы.
11.	2	Классы. Создания классов. Наследование, встраивание и полиморфизм	Вопросы 26-29. Выполнить задания 2 и описать в терминах классов.	Опрос. Защита работы
12.	2	Отношения между классами	Вопросы 30-31. Реферат. Выполнение задания 7	Защита реферата. Защита работы
13.	2	Объектные модели	Вопросы 32-37. Презентация	Опрос. Выступление

4.2 Характеристика заданий для самостоятельной работы обучающихся и методические рекомендации по их выполнению;

Для выполнения задания, прежде всего, необходимо ознакомиться и изучить основные положения теоретических материалов соответствующей темы из литературных источников. Они указаны в разделе «Содержание и структура дисциплины». Конспекты и задания можно выполнить в отдельном тетради или в лекционной (практической) тетради в произвольной форме.

4.3. Требования к представлению и оформлению результатов самостоятельной работы;

1. Индивидуальные домашние задания по самостоятельной работе должны быть выполнены в отдельной тетрадке. В каждом задании должны быть приведены постановка задачи и описана последовательность ее решения. В конце решения задачи приводятся результаты выполненной работы.
2. При выполнении самостоятельной работы студент должен предварительно изучить методы решения задач данного типа и правильно выбрать соответствующий метод ее решения.
3. По лабораторным работам студенты должны представить отчеты в соответствии с содержанием, приведенным в пункте 4.2, которые должны быть защищены у преподавателя. На защите лабораторных работ студентам задается один теоретический вопрос и задача, которые он должен самостоятельно подготовить и решить.

4.4. Критерии оценки выполнения самостоятельной работы.

4. Самостоятельная работа прививает студентам навыки работы с источниками и учебной литературой, помогает повысить уровень знаний по предмету, которые можно использовать на практике.

5. Оценка «отлично» выставляется студенту, если индивидуальное задание выполнено полностью и по данной теме защищена лабораторная работа.
6. Оценка «хорошо» выставляется студенту, если лабораторная работа по теме индивидуального задания защищена, а само индивидуальное задание выполнено с отдельными замечаниями.
7. Оценка «удовлетворительно» выставляется студенту, если лабораторная работа по теме индивидуального задания защищена, а само индивидуальное задание выполнено не до конца, т.е. не полностью.
8. Оценка «неудовлетворительно» выставляется студенту, если лабораторная работа по теме индивидуального задания не защищена, а само индивидуальное задание выполнено не до конца, т.е. не полностью.

5. СПИСОК УЧЕБНОЙ ЛИТЕРАТУРЫ И ИНФОРМАЦИОННО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

5.1. Основная литература

1. Грекул В. И., Коровкина Н. Л., Левочкина Г. А. Основы проектирования информационных систем (учебник) Национальный исследовательский университет «Высшая школа экономики» (г. Москва). 2021 385 стр.
2. Черткова Е. А. Введение в программную инженерию; Инженерия ПО Национальный исследовательский университет «Высшая школа экономики» (г. Москва). 2021 147 стр
3. ИркаевБ.Н., Умаров М.А., Бахтеев К.С. Основы информационных технологий, Учебник, РТСУ, Душанбе, 2018, 370 стр.
4. Кудрина Е. В., Огнева М. В. Основы алгоритмизации и программирования на языке C#. Национальный исследовательский Саратовский государственный университет имени Н.Г. Чернышевского (г. Саратов). УМО ВО 2021 286 стр.
5. Огнева М. В., Кудрина Е. В. ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ C++: ПРАКТИЧЕСКИЙ КУРС. Учебное пособие для вузов УМО ВО 2021 335 с.
6. Маркин А. В. ПРОГРАММИРОВАНИЕ НА SQL В 2 Ч. ЧАСТЬ 2 2-е изд., испр. и доп. Учебник и практикум для вузов УМО ВО 2021 340 с.
7. Григорьев М. В., Григорьева И. И. Методы и средства проектирования информационных систем и технологий; Тюменский государственный университет (г. Тюмень). 2021 318 стр.
8. Умаров М.А., Бахтеев К.С., Мирзокаримов О.А, «Проектирование информационных

5.2 Дополнительная литература

9. Умаров М.А. Проектирование информационных систем. Часть 1. Методологические основы проектирования информационных систем. Учебное пособие. Душанбе: - РТСУ, 2011. 125с.
10. Брауде Э.Дж., Технология разработки программного обеспечения. – СПб.: Питер, 2004. – 655с.
11. Бобровский С.И. Delphi 7. Учебный курс – СПб.: Питер, 2004. – 736с.
12. Буч Г., Максимчук Р., Энгл М. и др. Объектно-ориентированный анализ и проектирование с примерами приложений, 3-е изд.: Пер. с англ.- М.: ООО «И.Д.Вильямс», 2008.-720с.
13. Буч Г., Рамбо Д., Джекобсон И. Язык UML. Руководство пользователя: Пер. с англ.- М.: ДМК, 2010. 432 с.
14. Информационные системы в экономике и управлении: Учебник / Под ред. Проф. В.В. Три- фонова. – М.: Высшее образование, 2009. – 480 с.
15. Гарнаев А. И др. Microsoft Office 2000. Разработка приложений. СПб.: BHV, 2000, 656 с. С ил
16. Вендров А.М. Проектирование программного обеспечения экономических информационно- ных систем. - М.2000
17. Ефимов Е.Н., Петрушкина С.М., Панферова Л.Ф., Хашиева Л.И. Информационные системы в экономике. - М: ИКЦ «МарТ» , 2004. – 352с.
18. Кватрани Т. Rational Rose 2000 и UML. Визуальное моделирование: Пер с англ. – М.: ДМК Пресс, 2001.-176с.
19. Ли И.Т., Умаров М.А., Методические рекомендации по выполнению дипломных проектов для специальности 010502 «Прикладная информатика (в экономике)» Душанбе: РТСУ, 2009.- 101с.
20. http:// www.citforum.ru – материалы сайта Сервер информационных технологий.
21. http://www.computer.org/tab/seprof/code.htm

5.4. Перечень ресурсов информационно-телекоммуникационной сети «Интернет»

22. http:// www.citforum.ru – материалы сайта Сервер информационных технологий.
23. http://www.makasin.info/system/files

5.5. Перечень информационных технологий и программного обеспечения

Используются лицензионное программное обеспечение ОС Windows -/11 и программное обеспечение открытого доступа (Open source), среды программирования (Denwer, CodeBlock, Dev_C++ и др.). Для разработки моделей проекта ИС используются CASE – средства: ERWin, Visual UML, Rational Rose и т.д.

6. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

Курс **«Программная инженерия»** направлен на формирование у студентов знаний и практических навыков проектирования, разработки, тестирования и документирования программных продуктов с использованием современных методологий и технологий.

В первом разделе рассматриваются общие методологии и подходы к проектированию программного обеспечения. Студенты изучают основные этапы разработки ПО и знакомятся с разными методами проектирования: структурным, объектно-ориентированным и функциональным программированием. Особое внимание уделяется современным подходам, таким как Rapid Application Development (RAD), а также декларативному и логическому программированию.

Второй раздел посвящен проектированию информационного обеспечения. Здесь изучаются структуры и модели данных, а также инструменты моделирования, такие как ER-диаграммы, отражающие сущности, связи и атрибуты. Рассматриваются CASE-средства, например, ERWin, используемые для поддержки проектирования.

Третий раздел раскрывает методы структурного и объектно-ориентированного проектирования. Излагаются основные принципы структурного подхода — иерархия, абстрагирование и модульность. Рассматривается функциональное моделирование с использованием SADT. Особое внимание уделяется объектно-ориентированному программированию: инкапсуляции, наследованию, полиморфизму, классам и объектам.

В четвертом разделе студенты знакомятся с объектными моделями (OMG, CORBA) и средами объектно-ориентированного программирования, такими как C++, Delphi и Smalltalk. Рассматриваются механизмы областей видимости, переопределения методов, наследования и полиморфизма.

Пятый раздел посвящен изучению языка UML и CASE-технологий. Рассматриваются цели стандартизации UML, основные виды диаграмм — Use Case, Class, Sequence, Activity, Statechart, Component, Deployment. Изучается применение UML для комплексного моделирования системы с различных точек зрения — логической структуры, взаимодействия процессов, компонентного представления. Обзор популярных CASE-средств, таких как Rational Rose и Microsoft Visual Modeler, помогает освоить практические инструменты моделирования.

Шестой раздел охватывает вопросы документирования и обеспечения качества программных продуктов. Рассматриваются стандарты технологической документации, комплекты документов на всех этапах жизненного цикла ПО — проектирование, тестирование, сопровождение. Студенты знакомятся с методами оценки качества и управления версиями.

Практическая часть курса включает лабораторные работы по проектированию с использованием CASE-средств и UML, самостоятельные задания и курсовой проект, который предусматривает подготовку проектной документации и её защиту.

7. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Для реализации дисциплины при кафедре информатики и ИТ РТСУ имеются 4 компьютерных классов. Для занятий используются лицензионное программное обеспечение ОС Windows -7/8/10 и программное обеспечение открытого доступа (Open source), среды программирования (Denwer, CodeBlock, Dev_C++ и др.). Для разработки моделей проекта ИС используются CASE – средства: ERWin, Visual UML, Rational Rose и т.д.

В Университете созданы специальные условия для обучающихся с ограниченными возможностями здоровья - специальные учебники, учебные пособия и дидактические материалы, специальные технические средства обучения коллективного и индивидуального пользования, предоставление услуг ассистента (помощника), оказывающего обучающимся необходимую техническую помощь, проведение групповых и индивидуальных коррекционных занятий, обеспечение доступа в здания организаций и другие условия, без которых невозможно или затруднено освоение дисциплины обучающимися с ограниченными возможностями здоровья.

Обучающимся с ограниченными возможностями здоровья предоставляются бесплатно специальные учебники и учебные пособия, иная учебная литература, а также обеспечивается:

- наличие альтернативной версии официального сайта организации в сети "Интернет" для слабовидящих;
- присутствие ассистента, оказывающего обучающемуся необходимую помощь;
- обеспечение выпуска альтернативных форматов печатных материалов (крупный шрифт или аудиофайлы);
- возможность беспрепятственного доступа обучающихся в учебные помещения, столовые, туалетные и другие помещения организации, а также пребывания в указанных помещениях (наличие пандусов, поручней, расширенных дверных проёмов, лифтов).

8. ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ИТОГАМ ОСВОЕНИЯ ДИСЦИПЛИНЫ И УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ

Промежуточная аттестации осуществляется: для зачета – контрольная работа и опрос. Экзамен проводится в форме тестирования. Защита курсового проекта: представляется пояснительная записка

и презентация выступления.

Текущий контроль студентов осуществляется путем защиты лабораторных работ, выполнения самостоятельного задания, обсуждения теоретических вопросов.

Контролирующие материалы по дисциплине содержат:

Контрольные вопросы и задания для текущего контроля знаний по дисциплине.

Тестовые задания для промежуточного контроля знаний по дисциплине;

Методические рекомендации и тематика курсового проектирования.

Также указаны критерии оценки курсового проекта.

Итоговая система оценок по кредитно-рейтинговой системе с использованием буквенных символов

Оценка по буквенной системе	**Диапазон соответствующих наборных баллов**	**Численное выражение оценочного балла**	**Оценка по традиционной системе**
A	10	95-100	Отлично
A-	9	90-94	
B+	8	85-89	Хорошо
B	7	80-84	
B-	6	75-79	
C+	5	70-74	Удовлетворительно
C	4	65-69	
C-	3	60-64	
D+	2	55-59	
D	1	50-54	
Fx	0	45-49	Неудовлетворительно
F	0	0-44	

Содержание текущего контроля, промежуточной аттестации, итогового контроля раскрываются в фонде оценочных средств, предназначенных для проверки соответствия уровня подготовки по дисциплине требованиям ФГОС ВО. ФОС по дисциплине является логическим продолжением рабочей программы учебной дисциплины. ФОС по дисциплине прилагается.