МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Воронежский государственный технический университет»

УТВЕРЖДАЮ

Декан факультета ЭМИТ

_____/ С.А. Баркалов___/

_____ 31.08.2023 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

«Программная инженерия»

Направление подготовки 27.03.03 Системный анализ и управление

Профиль Бизнес-аналитика и системы больших данных

Квалификация выпускника бакалавр

Нормативный период обучения 4 года

Форма обучения очная

Год начала подготовки 2023

Автор программы _____ О.В. Минакова

И.О. зав. кафедрой Систем управления и информационных технологий в строительстве _____ Н.Г. Аснина

Руководитель ОПОП _____ О.С. Перевалова

Воронеж 2023

1. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ

1.1. Цели дисциплины формирование навыков автоматизация аналитических исследований с использованием инструментальных средств и фреймфорков

1.2. Задачи освоения дисциплины

- понимание требований пользователей к функциональности, удобству использования, обеспечению высокой доступности и безопасности систем анализа данных;
- изучение принципов построения современных программных инструментов анализа, интерпретации, построения отчётов и визуализации данных;
- получение опыта работы с проектами внедрению аналитических моделей в системы управления

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОПОП

Дисциплина «Программная инженерия» относится к дисциплинам части, формируемой участниками образовательных отношений блока Б1.

3. ПЕРЕЧЕНЬ ПЛАНИРУЕМЫХ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ

Процесс изучения дисциплины «Программная инженерия» направлен на формирование следующих компетенций:

ПК-2 - Способен проводить аналитические исследования с помощью методов системного анализа в соответствии с требованиями заказчика

ПК-3 - Способен осуществлять проектирование систем управления среднего и крупного масштаба и сложности

Компетенция	Результаты обучения, характеризующие сформированность компетенции
ПК-2	знать язык программирования, стандарты и паттерны разработки, основные архитектуры SOA, REST, MVC
	уметь проводить анализ требований и оценивать сложность проекта, работать с инструментами тестирования ПО, разрабатывать программные прототипы
	владеть навыками интеграции внешних систем и API в свои проекты, методами логгирования и мониторинга программных систем
ПК-3	знать методологии разработки программного обеспечения, принципы проектирования и архитектуры программных систем, методы тестирования и программирования
	уметь работать в среде разработки программного

	обеспечения, управлять сборкой, конфигурациями программной системы
	владеть методами анализа и оптимизации производительности ПО, тестирования, управления проектами и документирования

4. ОБЪЕМ ДИСЦИПЛИНЫ

Общая трудоемкость дисциплины «Программная инженерия» составляет 5 з.е.

Распределение трудоемкости дисциплины по видам занятий

очная форма обучения

Виды учебной работы	Всего часов	Семестры	
		6	7
Аудиторные занятия (всего)	104	32	72
В том числе:			
Лекции	52	16	36
Практические занятия (ПЗ)	18	-	18
Лабораторные работы (ЛР)	34	16	18
Самостоятельная работа	40	4	36
Курсовой проект	+		+
Часы на контроль	36	36	-
Виды промежуточной аттестации - экзамен, зачет с оценкой	+	+	+
Общая трудоемкость: академические часы зач.ед.	 180 5	 72 2	 108 3

5. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

5.1 Содержание разделов дисциплины и распределение трудоемкости по видам занятий

очная форма обучения

№ п/п	Наименование темы	Содержание раздела	Лекц	Прак зан.	Лаб. зан.	СРС	Всего, час
1	Основные принципы и методологии разработки ПО	Понятие технологии программирования как разработки надежного ПО. Проблемы разработки программного обеспечения. Понятие технологии и методологии. Жизненный цикл ПО: Стандарты в области программной инженерии. Основные и вспомогательные процессы разработки ПО. Стадии разработки ПО. Каскадная и эволюционная модель. Прототипирование. Итерационная и инкрементная разработка. Спиральная модель и оценка рисков. Современные методологии разработки ПО: Тяжеловесные и легковесные процессы. RUP, MFC. XP-программирование.	10	2	6	6	24

		Angle-техники: Scrum, Kanban. Понятие среды разработки (IDE, SDK) и CASE-средства. Механизмы интеграции инструментальных средств.					
2	Архитектурное проектирование По	Архитектурное проектирование: Архитектура с общим репозиторием, клиент-серверная, многоуровневая. Модели централизованного и событийного управления. Архитектура канала и фильтра. Декомпозиция программной архитектуры: Абстракция и декомпозиция. Связанность и сцепление. Пошаговая детализация. Иерархия программных модулей/функций. Объектная декомпозиция системной архитектуры. Основные понятия ООП. SOLID. Разработка и оценка архитектуры на основе сценариев. Виды диаграмм UML	10	2	6	6	24
3	Практики конструирования ПО	Основные приемы программирования: Механизмы абстракции. Типы и структуры данных. Управляющие конструкции. Рекурсия, структурная декомпозиция, функции и передача параметров. Обработка исключений и поиск ошибок. Использование утверждений Техники логгирования, интерпретации и анализа логгов, настройка инструментов логгирования API принципы построения и использования. Особенности и характеристики REST, SOAP	8	2	6	6	22
4	Тестирование ПО	Тестирование – как инструмент оценки ПО Тестирование на различных этапах жизненного цикла. Тестирование "белого ящика" на стадии кодирования. Способы тестирования базового пути, тестирования условий, циклов. Тестирование "черного ящика". Техники unit-тестирвоания,использование Selenium для автоматизации тестирваоиня	8	4	6	6	24
5	Управление выпуском и конфигурацией ПО	Модификация ПО: Эволюция программной архитектуры. Свойство ПО, пригодного для сопровождения. Особенности сопровождения программных продуктов. Унаследованные системы и реверсная инженерия. Конфигурационное управление. Принципы и средства управления конфигурацией. Управления версиями. Основы git. Инструменты совместной разработки и управления конфигурацией. Системы сборки и непрерывной интеграции Основы Docker. Создание докер-контейнера.	8	4	6	8	26
6	Оценка качества и рефакторинг ПО	Характеристики качества ПО: Модель качества по ISO 9126. Характеристики	8	4	4	8	24

		и субхарактеристики качества программного средства. Метрики качества программного средства. Оценивание характеристик качества программных средств. Практика составления спецификации качества Проведение оценки качества. Понятие рефакторинга. Принципы и паттерны рефакторинга. Инструменты Документирование ПО: Стандарты разработки программной документации.					
		Итого	**52**	**18**	**34**	**40**	**144**

5.2 Перечень лабораторных работ

1. Знакомство со средой разработки (IDE) и разработка консольного приложения небольшого размера
2. Создание простого GUI приложения с вводом данных
3. Обработка событий таймера
4. Работа с массивами
5. Работа со связанным списком
6. Работа с файлами
7. Работа с инструментами сборки программного проекта
8. Разработка программы построения графиков
9. Тестирование программного модуля (unit-тестирование)
10. Интеграционное тестирование
11. Реализация графического интерфейса к БД
12. Работа со сторонним API
13. Разработка html-редактора
14. Разработка клиент-серверного приложения
15. Логгирование с использование s4

6. ПРИМЕРНАЯ ТЕМАТИКА КУРСОВЫХ ПРОЕКТОВ (РАБОТ) И КОНТРОЛЬНЫХ РАБОТ

В соответствии с учебным планом освоение дисциплины предусматривает выполнение курсового проекта в 7 семестре для очной формы обучения.

Примерная тематика курсового проекта: «Разработка приложения анализа данных»

Задачи, решаемые при выполнении курсового проекта:

• Анализ предметной области и существующих программных решений, формулировка требований к ПО, написание технического задания;

• Проектирование программной системы, создание пользовательского интерфейса, моделирование последователи взаимодействия пользователя с системой

• Реализация программы, оформление кода и документации к программе

Курсовой проект включат в себя создание программного продукта и

оформление расчетно-пояснительной записки.

7. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

7.1. Описание показателей и критериев оценивания компетенций на различных этапах их формирования, описание шкал оценивания

7.1.1 Этап текущего контроля

Результаты текущего контроля знаний и межсессионной аттестации оцениваются по следующей системе:

«аттестован»;

«не аттестован».

Компе-тенция	Результаты обучения, характеризующие сформированность компетенции	Критерии оценивания	Аттестован	Не аттестован
ПК-2	знать язык программирования, стандарты и паттерны разработки, основные архитектуры SOA, REST, MVC	Тест	Выполнение теста на 70-100%	Выполнение менее 70%
	уметь проводить анализ требований и оценивать сложность проекта, работать с инструментами тестирования ПО, разрабатывать программные прототипы	Решение стандартных практических задач	Продемонстрирова н верный ход решения в большинстве задач	Задачи не решены
	владеть навыками интеграции внешних систем и API в свои проекты, методами логгирования и мониторинга программных систем	Решение прикладных задач в конкретной предметной области	Продемонстрирова н верный ход решения в большинстве задач	Задачи не решены
ПК-3	знать язык программирования, стандарты и паттерны разработки, основные архитектуры SOA, REST, MVC	Тест	Выполнение теста на 70-100%	Выполнение менее 70%
	уметь проводить анализ требований и оценивать сложность проекта, работать с инструментами тестирования ПО, разрабатывать программные прототипы	Решение стандартных практических задач	Продемонстрирова н верный ход решения в большинстве задач	Задачи не решены
	владеть навыками интеграции внешних систем и API в свои проекты, методами логгирования и мониторинга программных систем	Решение прикладных задач в конкретной предметной области	Продемонстрирова н верный ход решения в большинстве задач	Задачи не решены

7.1.2 Этап промежуточного контроля знаний

Результаты промежуточного контроля знаний оцениваются в 6, 7 семестре для очной формы обучения по четырехбалльной системе:

«отлично»;

«хорошо»;

«удовлетворительно»;

«неудовлетворительно».

Компе-тенция	Результаты обучения, характеризующие сформированность компетенции	Критерии оценивания	Отлично	Хорошо	Удовл.	Неудовл.
ПК-2	знать язык программирования, стандарты и паттерны разработки, основные архитектуры SOA, REST, MVC	Тест	Выполнение теста на 90-100%	Выполнение теста на 80-90%	Выполнение теста на 70-80%	В тесте менее 70% правильных ответов
	уметь проводить анализ требований и оценивать сложность проекта, работать с инструментами тестирования ПО, разрабатывать программные прототипы	Решение стандартных практических задач	Задачи решены в полном объеме и получены верные ответы	Продемонстрирован верный ход решения всех, но не получен верный ответ во всех задачах	Продемонстрирован верный ход решения в большинстве задач	Задачи не решены
	владеть навыками интеграции внешних систем и API в свои проекты, методами логгирования и мониторинга программных систем	Решение прикладных задач в конкретной предметной области	Задачи решены в полном объеме и получены верные ответы	Продемонстрирован верный ход решения всех, но не получен верный ответ во всех задачах	Продемонстрирован верный ход решения в большинстве задач	Задачи не решены
ПК-3	знать язык программирования, стандарты и паттерны разработки, основные архитектуры SOA, REST, MVC	Тест	Выполнение теста на 90-100%	Выполнение теста на 80-90%	Выполнение теста на 70-80%	В тесте менее 70% правильных ответов
	уметь проводить анализ требований и оценивать сложность проекта, работать с инструментами тестирования ПО, разрабатывать программные прототипы	Решение стандартных практических задач	Задачи решены в полном объеме и получены верные ответы	Продемонстрирован верный ход решения всех, но не получен верный ответ во всех задачах	Продемонстрирован верный ход решения в большинстве задач	Задачи не решены
	владеть навыками интеграции внешних систем и API в свои проекты, методами логгирования и	Решение прикладных задач в конкретной предметной	Задачи решены в полном объеме и получены	Продемонстрирован верный ход решения всех, но не получен	Продемонстрирован верный ход решения в большинстве	Задачи не решены

	мониторинга программных систем	области	верные ответы	верный ответ во всех задачах	задач	

7.2 Примерный перечень оценочных средств (типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности)

7.2.1 Примерный перечень заданий для подготовки к тестированию

1. Выберите правильное утверждение

a) Программный модуль – это любой фрагмент кода, который программируется, компилируется и отлаживается отдельно

b) Программные модули основа объектно-ориентированного программирования

c) Программный модуль состоит из одного объекта или функции

d) Каждая функция программы состоит из отдельных модулей

2. Укажите оптимальный тестовый набор для значения x лежащего в диапазоне от 0 до 10

a) -1, 5, 11

b) 0, 1, 2, ...,10

c) 0, 10

d) 2,8, 11

e) 17

3. Принцип абстрагирования заключается в

a) выделение существенных аспектов системы и отвлечение от несущественных;

b) сложных проблемы путем их разбивки на множество меньших независимых задач

c) организации составных частей проблемы в иерархические древовидные структуры

d) в обоснованности и согласованности элементов;

4. Методы класса определяют:

a) какие операции можно выполнять с объектами данного класса

b) какие типы значений могут принимать данные-члены класса

c) каким образом реализуется защита данных-членов класса

d) как изменяется сущность

5. Структура данных, обеспечивающая быструю вставку и очень быстрый доступ, если известен индекс

a) Массив

b) Стек

c) Очередь

d) Хеш-таблица

6. Из представленных языковых конструкций укажите объявление переменной:

a) int a;

b) a=5;

c) return;

d) double countIMT (int a);

8. Принцип «иерархического упорядочивания» заключается

a) в том, что данные должны быть структурированы и иерархически организованы

b) организации составных частей проблемы в иерархические древовидные структуры

c) выделение существенных аспектов системы и отвлечение от несущественных;

d) в обоснованности и согласованности элементов

9. Отладка - это

a) Исправление ошибок в коде

b) Поиск ошибок

c) Проверка спецификации

d) Выполнение теста

10. Событие OnClick определяет реакцию программы на:

a) щелчок указателя мыши над компонентом

b) двойной щелчок указателя мыши над компонентом

c) перемещение указателя мыши над компонентом

d) получение фокуса ввода

7.2.2 Примерный перечень заданий для решения стандартных задач

1. Для программной системы «Метеостанция» составьте иерархию заинтересованных лиц и их ролей, используя метод опорных точек.

2. Для программной системы «Библиотека ВУЗа» смоделируйте для каждого выделенного действующего лица варианты использования.

3. Составьте перечень пользовательских и функциональных требований к программе «Графический редактор».

4. Составьте лист оценки характеристик качества программных средств построения UML-диаграмм и проведение экспертную оценки 3-4 программных продуктов.

5. Для любой ранее разработанной программы создайте механизм защиты полей ввода, включающий:

- *Контроль границ допустимых значений;*
- *«Эхо»;*
- *Обработка исключений.*

6. Оформите программный код с использованием JavaDoc

7. Выполните анализ кода программы в соответствии с рекомендациями Code Conventions for the Java TM Programming Language

8. Подготовить кейс-тесты для проверки функций программы анализа текста, со следующими функциональными требованиями:

Программа осуществляет чтение файла input.txt и выводит в файл output.txt следующую информацию:

- кол-во букв (1), знаков препинания (2) и цифр(3) в заданном тексте;*

- кол-во слов (4) в заданном тесте;

- кол-во предложений (5) в заданном тесте

в виде числовых значений на разных строках в указанном порядке (в скобках - позиция в файле)

9. Разработать программную документацию на разработанный программный продукт

10. Для класса:class Complex {

```
private final double re;
private final double im;

public Complex(double re, double im) {
    this.re = re;
    this.im = im;
}
Complex divide(Complex c) {
    double d = c.re * c.re + c.im * c.im;
    return new Complex(
        (re * c.re + im * c.im) / d,
        (im * c.re - re * c.im) / d
    );
}
```

a) Предложите дисциплинированную обработку ошибок (не обработка исключений)

b) Представьте пример использования обработки исключений в реализации одного из методов

c) Добавьте возможность выбрасывание собственного исключения в этот класс

d) Придумайте примеры реализации определения этого класса для трех видов гарантий безопасности.

7.2.3 Примерный перечень заданий для решения прикладных задач

1. Разработать и реализовать класс Student: Фамилия, Имя, Отчество, Дата рождения, Адрес, Телефон, Факультет, Курс.

Создать массив объектов.

Вывести:

а) список студентов заданного факультета;

б) списки студентов для каждого факультета и курса;

в) список студентов, родившихся после заданного года

2. Разработать и реализовать класс Abiturient: Фамилия, Имя, Отчество, Адрес, Баллы по предметам.

Создать массив объектов.

Вывести:

а) список абитуриентов, имеющих неудовлетворительные оценки;

б) список абитуриентов, сумма баллов у которых не меньше заданной;

в) выбрать N абитуриентов, имеющих самую высокую сумму баллов, и список абитуриентов, имеющих полупроходной балл.

3. Разработать и реализовать класс Apartment: Адрес, Этаж, Количество комнат, Площадь.

Создать массив объектов.

Вывести:

а) список квартир, имеющих заданное число комнат;

б) список квартир, имеющих заданное число комнат и расположенных на этаже, который находится в определенном промежутке;

в) список квартир, имеющих площадь, превосходящую заданную.

4. Определить объект файл с поддержкой операций создания, копирования, перемещения, удаления файла, наполнения содержимым. Унаследовать от него класс «Зашифрованный файл» с поддержкой функций шифрации – дешифрации содержимого. В программе продемонстрировать функциональность разработанных классов.

5. Определить абстрактный класс Currency для работы с денежными суммами. Определить в нем методы перевода в рубли и вывода на экран. На его основе реализовать классы Dollar, Euro и Pound (фунт стерлингов) с возможностью пересчета в центы и пенсы соответственно и указанием текущего курса.. Создать класс Purse (кошелек), содержащий массив объектов этих классов в динамической памяти. Предусмотреть возможность случайного наполнения кошелька купюрами различного типа и подсчета общей суммы кошелька при изменении курса валют. В программе продемонстрировать функциональность разработанных классов.

6. Определить абстрактный класс Function (функция) с виртуальными методами вычисления значения функции y = f(x) в заданной точке x и вывода результата на экран, поиска минимума и максимума функции на заданном интервале.. Унаследовать от класса Function классы Hiperbola и Parabola, Exponenta, в которых уточняется функция f(x) и задаются коэффициенты соответствующих функций. Продемонстрировать функциональные возможности этих классов (получением значений, вычислением максимума или минимума).

7. Разработайте и реализуйте графический интерфейс пользователя для следующей системы:

1) Обработка информации по оказанию услуг ЖКХ населению.

2) Обработка информации по списанию основных средств.

3) Формирование цены продажи продукции.

4) Обработка информации по договорам с контрагентами.

5) Учет выплаты стипендии студентам.

8. Спроектируйте графический интерфейс и реализуйете компьютерную игру Реверси

9. Постройте UML-диаграммы для программной реализации программной системы «Обработка информации по расчетам с поставщиками»

10. Обоснуйте выбор программной архитектуры и разработайте проектную документацию для программной системы «Расчет рейтинга студентов».

7.2.4 Примерный перечень вопросов для подготовки к зачету

Не предусмотрено учебным планом

7.2.5 Примерный перечень заданий для подготовки к экзамену

1. Цели и задачи технологий разработки ПО. Особенности современных крупных проектов разработки ПО.

2. Понятие программная инженерия. Основные, вспомогательные и организационные процессы программной инженерии.

3. Структурный подход к проектированию ПО. Сущность структурного подхода.

4. Объектно-ориентированная разработка программ. Объектно-ориентированные языки программирования. Объектно-ориентированные методологии разработки программных систем.

5. Каскадная модель жизненного цикла ПС: содержание этапов, область применения, достоинства и недостатки.

6. Эволюционная модель жизненного цикла ПС: последовательность действий, область применения, достоинства и недостатки.

7. Спиральная модель разработки ПО: содержание этапов создания ПС, область применения, достоинства и недостатки.

8. Инкрементальная модель разработки ПО. Развитие инкрементального подхода. XP-процессы.

9. Понятие программного проекта. Управление программным проектом. План и содержание его разделов. Составление сетевого графика работ.

11. Состав и структура коллектива разработчиков программного продукт, их функции. Составление расписания (PERT-диаграммы)

12. Управление документацией разработки программного продукта.

13. Рациональный Унифицированный Процесс. Динамические аспекты процессов: структура ЖЦ, стадии, итерации и контрольные точки.

14. Рациональный Унифицированный Процесс. Статическое содержание процесса: виды деятельности (технологические операции), рабочие продукты, исполнители и дисциплины (технологические процессы).

15. Внешнее описание программного средства и спецификация. Виды требований к ПО: системные, функциональные, характеристики качества.

16. Методы определения и формализация требований к ПО.

17. Понятие качества ПО и его многоуровневая модель. Характеристики и атрибуты качества.

18. Разработка требований к ПО: формирование и анализ, документирование, аттестация. Управление.

19. Алгоритмическая декомпозиция. Модульное программирование. Характеристики программного модуля.

20. Модели архитектур с различными способами обмена данными: репозиторий, «клиент-сервер».

21. Архитектуры с различными моделями управления.

22. Событийно-управляемые архитектуры.

23. Модели архитектур с различными подходами к обработке данных: непрерывная обработка, каналы и фильтры.

24. Объектно-ориентированная декомпозиция. Общая

характеристика объектов. Виды отношений между объектами. Агрегация.

25. Абстрагирование. Общая характеристика классов. Виды отношений между классами. Ассоциации классов. Наследование. Полиморфизм. Агрегация.

26. Повторное использование компонентов. Инкапсуляция. Интерфейсы. Компонентная объектная модель (COM).

27. Принципы проектирования пользовательского интерфейса.

28. Структурное тестирование. Покрытие операторов, ветвей, условий.

29. Функциональное тестирование. Метод эквивалентного разбиения, граничных значений, причинно-следственных (функциональных) диаграмм.

30.Тестирование интеграции компонентов ПО: нисходящее и восходящее. Понятие драйвер и заглушка. Стохастическое тестирование.

31. Разработка программной документации. С-документация и П-документация.

32.Отладка ПО: цели и методы.

33. Управление конфигурацией ПО. Системы контроля версий. Регрессионное тестирование.

34. Аттестация ПО. Оценка качества ПО.

35. Инструментальные средства разработки ПО. Автоматизация разработки ПО. CASE-средства.

36. Сопровождение ПО. Основные подходы: с целью исправления ошибок, адаптации и изменения функциональных возможностей. Решение проблемы эволюции ПО – рефакторинг, реинженерия, реверсная инженерия

7.2.6. Методика выставления оценки при проведении промежуточной аттестации

Экзамен проводится по тест-билетам, каждый из которых содержит 30 вопросов. Каждый правильный ответ на вопрос в тесте оценивается 1 баллом. Максимальное количество набранных баллов – 30.

1. Оценка «Неудовлетворительно» ставится в случае, если студент набрал менее 14 баллов.

2. Оценка «Удовлетворительно» ставится в случае, если студент набрал от 15 до 20 баллов

3. Оценка «Хорошо» ставится в случае, если студент набрал от 21 до 25 баллов.

4. Оценка «Отлично» ставится, если студент набрал 26 баллов и выше

7.2.7 Паспорт оценочных материалов

№ п/п	Контролируемые разделы (темы) дисциплины	Код контролируемой компетенции	Наименование оценочного средства
1	Основные принципы и методологии разработки ПО	ПК-2, ПК-3	Тест, практическое задание, защита лабораторных работ, требования к курсовому проекту
2	Архитектурное проектирование По	ПК-2, ПК-3	Тест, практическое задание, защита лабораторных работ, требования к

			курсовому проекту
3	Практики конструирования ПО	ПК-2, ПК-3	Тест, практическое задание, защита лабораторных работ, требования к курсовому проекту
4	Тестирование ПО	ПК-2, ПК-3	Тест, практическое задание, защита лабораторных работ, требования к курсовому проекту
5	Управление выпуском и конфигурацией ПО	ПК-2, ПК-3	Тест, практическое задание, защита лабораторных работ, требования к курсовому проекту
6	Оценка качества и рефакторинг ПО	ПК-2, ПК-3	Тест, практическое задание, защита лабораторных работ, требования к курсовому проекту

7.3. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

Тестирование осуществляется при помощи компьютерной системы тестирования, автоматически генерирующий 13 вопросов и базы по заданной теме. Время тестирования 15 мин. Более 6 правильных ответов интерпретируется как аттестован.

Решение стандартных задач осуществляется при помощи компьютерной системы тестирования. Время решения задач 30 мин. Затем осуществляется проверка решения задач экзаменатором и выставляется оценка, согласно методики выставления оценки при проведении промежуточной аттестации.

Решение прикладных задач осуществляется с использованием выданных задач. Задача решается в индивидуальном темпе в течение 2 недели. Затем осуществляется проверка решения задач экзаменатором и выставляется оценка, согласно методики выставления оценки при проведении промежуточной аттестации.

Защита курсовой работы, курсового проекта или отчета по всем видам практик осуществляется согласно требованиям, предъявляемым к работе, описанным в методических материалах. Примерное время защиты на одного студента составляет 20 мин.

8 УЧЕБНО МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ)

8.1 Перечень учебной литературы, необходимой для освоения дисциплины

1. Кознов, Д. В. Введение в программную инженерию : учебное пособие / Д. В. Кознов. — 3-е изд. — Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. — 305 с. — ISBN 978-5-4497-0311-8. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/89428.html (дата обращения: 18.03.2023). — Режим доступа: для авторизир. пользователей

2. Программная инженерия. Ч.III : курс лекций / составители Т. В. Киселева. — Ставрополь : Северо-Кавказский федеральный университет, 2018. — 130 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/92584.html (дата обращения: 18.03.2023). — Режим доступа: для авторизир. Пользователей

3. Джонс, М. Т. Программирование искусственного интеллекта в приложениях / М. Т. Джонс ; перевод А. И. Осипов. — 2-е изд. — Саратов : Профобразование, 2019. — 312 c. — ISBN 978-5-4488-0116-7. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/89866.html (дата обращения: 18.03.2023). — Режим доступа: для авторизир. пользователей

8.2 Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень лицензионного программного обеспечения, ресурсов информационно-телекоммуникационной сети «Интернет», современных профессиональных баз данных и информационных справочных систем:

Программное обеспечение

Среда разработки VS Code

Информационно-справочные системы

1. Сайт habr.com - аналитические обзоры по IT

2. Сайт CITForum www.citforum.ru Библиотека технических материалов по информационным технологиям

3. Сайты поддержки Python-разработчиков https://www.python.org/

4. Программная инженерия http://www.software-engin.com/ http://www.cs.st-andrews.ac.uk Авторские обзоры по современным тенденциям в инженерии ПО, обновление глав учебника «Программная инженерия»

9 МАТЕРИАЛЬНО-ТЕХНИЧЕСКАЯ БАЗА, НЕОБХОДИМАЯ ДЛЯ ОСУЩЕСТВЛЕНИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА

Для проведения лекционных занятий используется компьютер с установленным программным обеспечением для демонстрации презентаций и проектор.

Лабораторные занятия проводятся в компьютерных классах с установленным программным обеспечением:

средой разработки программного обеспечения;

инструментальными средствами git, maven

10. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ (МОДУЛЯ)

По дисциплине «Программная инженерия» читаются лекции, проводятся практические занятия и лабораторные работы, выполняется курсовой проект.

Основой изучения дисциплины являются лекции, на которых излагаются наиболее существенные и трудные вопросы, а также вопросы, не

нашедшие отражения в учебной литературе.

Практические занятия направлены на приобретение практических навыков работы с инструментами разработки ПО. Занятия проводятся путем решения прикладных проблем работы с программным обеспечением в аудитории.

Лабораторные работы выполняются в среде разработки программного обеспечения с использованием специальных инструментов разработки в соответствии с методиками, приведенными в указаниях к выполнению работ.

Методика выполнения курсового проекта изложена в учебно-методическом пособии. Выполнять этапы курсового проекта должны своевременно и в установленные сроки.

Контроль усвоения материала дисциплины производится проверкой курсового проекта, защитой курсового проекта.

Вид учебных занятий	Деятельность студента
Лекция	Написание конспекта лекций: кратко, схематично, последовательно фиксировать основные положения, выводы, формулировки, обобщения; помечать важные мысли, выделять ключевые слова, термины. Проверка терминов, понятий с помощью энциклопедий, словарей, справочников с выписыванием толкований в тетрадь. Обозначение вопросов, терминов, материала, которые вызывают трудности, поиск ответов в рекомендуемой литературе. Если самостоятельно не удается разобраться в материале, необходимо сформулировать вопрос и задать преподавателю на лекции или на практическом занятии.
Практическое занятие	Конспектирование рекомендуемых источников. Работа с конспектом лекций, подготовка ответов к контрольным вопросам, просмотр рекомендуемой литературы. Прослушивание аудио- и видеозаписей по заданной теме, выполнение расчетно-графических заданий, решение задач по алгоритму.
Лабораторная работа	Лабораторные работы позволяют научиться применять теоретические знания, полученные на лекции при решении конкретных задач. Чтобы наиболее рационально и полно использовать все возможности лабораторных для подготовки к ним необходимо: следует разобрать лекцию по соответствующей теме, ознакомится с соответствующим разделом учебника, проработать дополнительную литературу и источники, решить задачи и выполнить другие письменные задания.
Самостоятельная работа	Самостоятельная работа студентов способствует глубокому усвоения учебного материала и развитию навыков самообразования. Самостоятельная работа предполагает следующие составляющие: - работа с текстами: учебниками, справочниками, дополнительной литературой, а также проработка конспектов лекций; - выполнение домашних заданий и расчетов; - работа над темами для самостоятельного изучения; - участие в работе студенческих научных конференций, олимпиад; - подготовка к промежуточной аттестации.
Подготовка к	Готовиться к промежуточной аттестации следует систематически,

промежуточной аттестации	в течение всего семестра. Интенсивная подготовка должна начаться не позднее, чем за месяц-полтора до промежуточной аттестации. Данные перед экзаменом, зачетом с оценкой три дня эффективнее всего использовать для повторения и систематизации материала.

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

№ п/п	Перечень вносимых изменений	Дата внесения изменений	Подпись заведующего кафедрой, ответственной за реализацию ОПОП