Автономная некоммерческая организация высшего образования
«Волгоградский институт бизнеса»

Рабочая программа учебной дисциплины

Программная инженерия

(Наименование дисциплины)

09.03.03 Прикладная информатика, направленность (профиль) «Прикладной искусственный интеллект»

(Направление подготовки / Профиль)

Бакалавр

(Квалификация)

Кафедра разработчик Экономики и управления

Год набора **2026**

Вид учебной деятельности	Трудоемкость (объем) дисциплины	
	Очная форма	Очно-заочная форма
	д	в
Зачетные единицы	**3**	**3**
Общее количество часов	108	108
Аудиторные часы контактной работы обучающегося с преподавателями:	32	16
– Лекционные (Л)	16	8
– Практические (ПЗ)	16	8
– Лабораторные (ЛЗ)		
– Семинарские (СЗ)		
Самостоятельная работа обучающихся (СРО)	40	56
К (Р-Г) Р (П) (+;-)		
Тестирование (+;-)		
ДКР (+;-)		
Зачет (+;-)		
Зачет с оценкой (+;- (Кол-во часов))		
Экзамен (+;- (Кол-во часов))	+(36)	+(36)

Волгоград 2026

Содержание

Раздел 1. Организационно-методический раздел

1.1. Цели освоения дисциплины

Дисциплина **«Программная инженерия»** входит **в часть, формируемую участниками образовательных отношений Б1.В.09** подготовки обучающихся по направлению **Прикладная информатика, направленность (профиль) «Прикладной искусственный интеллект».**

Целью дисциплины является формирование **компетенций** (*в соответствии с ФГОС ВО и требованиями к результатам освоения основной профессиональной образовательной программы высшего образования (ОПОП ВО)*):

ОПК-6. Способен анализировать и разрабатывать организационно-технические и экономические процессы с применением методов системного анализа и математического моделирования;

Дескрипторы общепрофессиональных компетенций:

ОПК-6.1 – Способен на основе методов системного анализа и математического моделирования осуществлять разработку бизнес-требований к системе, включая анализ целесообразности применения методов искусственного интеллекта, формирование требований к данным и метрикам качества моделей.

ОПК-6.2 – Способен на основе методов системного анализа и математического моделирования выполнять постановку целей, разработку концепции системы, разработку технического задания на создание, в том числе для систем, использующих технологии искусственного интеллекта, с учетом особенностей жизненного цикла ИИ-моделей.

ПК-2. Способен разрабатывать технические спецификации на программные компоненты и их взаимодействие

Дескрипторы профессиональных компетенций:

ПК-2.1. Способен формализовывать требования и разрабатывать технические спецификации программных компонентов, включая компоненты анализа данных и искусственного интеллекта

ПК-2.2. Способен разрабатывать и согласовывать спецификации взаимодействия программных компонентов и интерфейсов, включая API интеллектуальных сервисов

Перечисленные компетенции формируются в процессе достижения **индикаторов компетенций:**

Обобщенная трудовая функция/ трудовая функция	**Код и наименование дескриптора компетенций**	**Код и наименование индикатора достижения компетенций (из ПС)**
	ОПК-6.1 – Способен на основе методов системного анализа и математического моделирования осуществлять разработку бизнес-требований к системе, включая анализ целесообразности применения методов искусственного интеллекта, формирование требований к данным и метрикам качества моделей. **ОПК-6.2** – Способен на основе методов системного анализа и математического моделирования вы-	Знает ИД-1 ОПК-6.1 Методы системного анализа и математического моделирования для разработки бизнес-требований к системе, включая анализ целесообразности применения методов искусственного интеллекта, формирование требований к данным и метрикам качества моделей (без привязки к профессиональному стандарту) ИД-2 ОПК-6.2 Подходы к постановке целей, разработке концепции системы и технического задания на создание, включая особенности жизненного цикла систем, использующих технологии искусственного интеллекта (без привязки к профессиональному стандарту) Умеет

	полнять постановку целей, разработку концепции системы, разработку технического задания на создание, в том числе для систем, использующих технологии искусственного интеллекта, с учетом особенностей жизненного цикла ИИ-моделей.	ИД-3 ОПК-6.1 Применять методы системного анализа и математического моделирования для разработки бизнес-требований к системе, включая оценку целесообразности применения ИИ, формирование требований к данным и метрикам качества моделей (без привязки к профессиональному стандарту) ИД-4 ОПК-6.2 Использовать методы системного анализа и математического моделирования для постановки целей, разработки концепции системы и технического задания, с учётом особенностей жизненного цикла ИИ-систем (без привязки к профессиональному стандарту) Имеет навыки ИД-5 ОПК-6.1 Владение навыками системного анализа и математического моделирования для разработки бизнес-требований, оценки целесообразности применения ИИ, формирования требований к данным и метрикам качества моделей (без привязки к профессиональному стандарту) ИД-6 ОПК-6.2 Владение навыками постановки целей, разработки концепции и технического задания на создание систем с использованием технологий искусственного интеллекта с учётом особенностей их жизненного цикла (без привязки к профессиональному стандарту)

ПС 06.001 Программист **D Разработка требований и проектирование программного обеспечения** **D/02.6.** Разработка технических спецификаций на программные компоненты и их взаимодействие	ПК-2.1. Способен формализовывать требования и разрабатывать технические спецификации программных компонентов, включая компоненты анализа данных и искусственного интеллекта ПК-2.2. Способен разрабатывать и согласовывать спецификации взаимодействия программных компонентов и интерфейсов, включая API интеллектуальных сервисов	Знает: ИД-1 ПК 2.1. Методы и приемы формализации задач D/02.6 ИД-2 ПК 2.2. Методы и средства проектирования программных интерфейсов D/02.6 Умеет: ИД-3 ПК 2.1. Проводить оценку и обоснование рекомендуемых решений D/02.6 ИД-4 ПК 2.2. Осуществлять коммуникации с заинтересованными сторонами D/02.6 Имеет навыки и (или) опыт: ИД-5 ПК 2.1. Осуществление контроля выполнения заданий D/02.6 ИД-6 ПК 2.2. Осуществление обучения и наставничества D/02.6

1.2. Место дисциплины в структуре ОПОП ВО
направления подготовки «09.03.03 Прикладная информатика», направленность (профиль) «Прикладной искусственный интеллект»

№	Предшествующие дисциплины (дисциплины, изучаемые параллельно)	Последующие дисциплины
1	*2*	*3*
1	Математическая логика	Основы систем искусственного интеллекта
2	Информатика и программирование	Проектирование систем с использованием технологий искусственного интеллекта
3	Автоматизированные информационные системы для бизнеса	Выполнение и защита выпускной квалификационной работы
4		Учебная практика (Научно-исследовательская работа (получение первичных навыков научно-исследовательской работы))
5		Производственная практика (Технологическая (проектно-технологическая) практика)
6		Производственная практика (Эксплуатационная практика)
7		Производственная практика (Преддипломная практика)

Последовательность формирования компетенций в указанных дисциплинах может быть изменена в зависимости от формы и срока обучения, а также преподавания с использованием дистанционных технологий обучения.

1.3. Нормативная документация

Рабочая программа учебной дисциплины составлена на основе:

- Федерального государственного образовательного стандарта высшего образования по направлению подготовки **09.03.03 Прикладная информатика**;
- Учебного плана направления подготовки **09.03.03 Прикладная информатика, направленность (профиль) «Прикладной искусственный интеллект»** 2026 года набора;
- Образца рабочей программы учебной дисциплины (приказ № 113-О от 01.09.2021 г.).

Раздел 2. Тематический план

Очная форма обучения (полный срок)

№	Тема дисциплины	Трудоемкость				
		Всего	Аудиторные занятия		СРО	Код индикатора и дескриптора достижения компетенций
			Л	ПЗ (ЛЗ, СЗ)		
1	*2*	*3*	*4*	*5*	*6*	*7*
1	Принципы ООП. Объектное проектирование. Структуры данных.	9	2	2	5	ИД-1 ОПК- 6.1 ИД-2 ОПК- 6.2
2	Алгоритмы и оценка их сложности	9	2	2	5	ИД-3 ОПК- 6.1 ИД-4 ОПК- 6.2
3	Методологии программной инженерии	9	2	2	5	ИД-5 ОПК- 6.1 ИД-6 ОПК- 6.2
4	Управление проектами разработки программного обеспечения	9	2	2	5	ИД-1 ПК- 2.1 ИД-2 ОПК- 2.2
5	Проектирование ПО для решения задач в области ИИ	9	2	2	5	ИД-3 ПК- 2.1 ИД-4 ПК- 2.2
6	Жизненный цикл моделей машинного обучения	9	2	2	5	ИД-3 ПК- 2.1 ИД-4 ОПК- 2.2
7	Основы промышленной разработки ПО	9	2	2	5	ИД-5 ПК- 2.1 ИД-6 ОПК- 2.2
8	Обеспечение качества и безопасность программного обеспечения	9	2	2	5	ИД-5 ПК- 2.1 ИД-6 ПК- 2.2
Вид промежуточной аттестации (Экзамен)		**+(36)**				
Итого		**108**	**16**	**16**	**40**	

Очно-заочная форма обучения (полный срок)

№	Тема дисциплины	Трудоемкость				
		Всего	Аудиторные занятия		СРО	Код индикатора и дескриптора достижения компетенций
			Л	ПЗ (ЛЗ, СЗ)		
1	*2*	*3*	*4*	*5*	*6*	*7*
1	Принципы ООП. Объектное проектирование. Структуры данных.	8	2		6	ИД-1 ОПК- 6.1 ИД-2 ОПК- 6.2
2	Алгоритмы и оценка их сложности	8	2		6	ИД-3 ОПК- 6.1 ИД-4 ОПК- 6.2
3	Методологии программной инженерии	8	2		6	ИД-5 ОПК- 6.1 ИД-6 ОПК- 6.2
4	Управление проектами разработки программного обеспечения	8	2		6	ИД-1 ПК- 2.1 ИД-2 ОПК- 2.2
5	Проектирование ПО для решения задач в области ИИ	10		2	8	ИД-3 ПК- 2.1 ИД-4 ПК- 2.2
6	Жизненный цикл моделей машинного обучения	10		2	8	ИД-3 ПК- 2.1 ИД-4 ОПК- 2.2
7	Основы промышленной разработки ПО	10		2	8	ИД-5 ПК- 2.1 ИД-6 ОПК- 2.2
8	Обеспечение качества и безопасность	10		2	8	ИД-5 ПК- 2.1

	программного обеспечения					ИД-6 ПК- 2.2
Вид промежуточной аттестации (Экзамен)		**+(36)**				
Итого		**108**	**8**	**8**	**56**	

Раздел 3. Содержание дисциплины

3.1. Содержание дисциплины

Тема 1. Принципы ООП. Объектное проектирование. Структуры данных.

Основные принципы объектно-ориентированного программирования: инкапсуляция (сокрытие данных, модификаторы доступа), наследование (иерархия классов, переопределение методов), полиморфизм (статический и динамический, виртуальные функции). Абстракция и интерфейсы как инструменты управления сложностью. Объектное проектирование: принципы SOLID (Single Responsibility, Open-Closed, Liskov Substitution, Interface Segregation, Dependency Inversion), их практическое значение для создания поддерживаемого и расширяемого кода. Паттерны проектирования: порождающие (Singleton, Factory, Abstract Factory), структурные (Adapter, Decorator, Facade), поведенческие (Observer, Strategy, Command). Структуры данных: линейные (массивы, списки, стеки, очереди), иерархические (деревья, бинарные деревья поиска), хеш-таблицы, графы. Выбор структур данных в зависимости от требований к производительности и характера операций. Связь между объектной моделью и структурами данных при проектировании программных систем.

Тема 2. Алгоритмы и оценка их сложности.

Понятие алгоритма: свойства (детерминированность, конечность, массовость, результативность). Способы описания алгоритмов. Оценка сложности алгоритмов: временная и ёмкостная (пространственная) сложность. Асимптотические обозначения: O (верхняя граница), Ω (нижняя граница), Θ (точная граница). Анализ сложности базовых алгоритмов: поиск (линейный, бинарный), сортировка (пузырьковая, быстрая, слиянием), обходы графов (DFS, BFS). Алгоритмы на графах: поиск кратчайших путей (Дейкстра, Флойда-Уоршелла), построение минимального остовного дерева (Прима, Краскала). Алгоритмы для работы со строками: поиск подстроки (Кнута-Морриса-Пратта, Бойера-Мура). Динамическое программирование: принцип оптимальности Беллмана, классические задачи (задача о рюкзаке, наибольшая общая подпоследовательность). Жадные алгоритмы: область применимости, свойства оптимальной подструктуры и жадного выбора. Практические аспекты выбора алгоритмов в зависимости от масштаба данных и ограничений ресурсов.

Тема 3. Методологии программной инженерии.

Классические методологии разработки: каскадная (Waterfall) модель — этапы, преимущества, ограничения, области применения. Итеративные модели: V-модель, спиральная модель (управление рисками). Гибкие (Agile) методологии: манифест Agile, ценности и принципы. Scrum: роли (Product Owner, Scrum Master, Development Team), артефакты (Product Backlog, Sprint Backlog, Increment), события (Sprint, Sprint Planning, Daily Scrum, Sprint Review, Sprint Retrospective). Kanban: принципы, визуализация потока задач, ограничение незавершённой работы (WIP), метрики (время выполнения, пропускная способность). Extreme Programming (XP): практики (парное программирование, непрерывная интеграция, тестирование, рефакторинг, коллективное владение кодом). Lean-разработка: устранение потерь, усиление обучения, отложенные решения. Сравнительный анализ методологий: критерии выбора в зависимости от типа проекта, размера команды, требований к документации и стабильности требований.

Тема 4. Управление проектами разработки программного обеспечения.

Основы управления проектами: фазы жизненного цикла проекта (инициация, планирование, исполнение, мониторинг, завершение). Формирование требований: сбор, анализ, документирование, управление изменениями. Методы оценки трудоёмкости: экспертные оценки, аналоговое

оценивание, метод PERT, story points в Agile. Планирование и декомпозиция работ: иерархическая структура работ (WBS), диаграммы Ганта, сетевые графики. Управление рисками: идентификация, анализ, планирование реагирования, мониторинг. Управление командой: мотивация, коммуникации, распределение ролей, разрешение конфликтов. Инструменты управления проектами: Jira, Trello, Asana, MS Project. Метрики проектной деятельности: скорость команды (velocity), диаграммы сгорания задач (burndown chart), диаграммы накопления задач (burnup chart). Управление качеством в проекте: планирование качества, контроль качества, улучшение процессов.

Тема 5. Проектирование ПО для решения задач в области ИИ.

Особенности проектирования систем искусственного интеллекта: недетерминированность поведения, зависимость от данных, сложность оценки качества. Архитектурные подходы: микросервисная архитектура для ИИ-компонентов, выделение сервисов инференса и обучения. Проектирование пайплайнов обработки данных: сбор данных, предобработка, валидация, хранение (feature store). Проектирование экспериментов: версионирование данных и моделей, воспроизводимость, логирование метрик (MLflow, Weights & Biases). Интерфейсы взаимодействия с ИИ-моделями: REST API, gRPC, асинхронная обработка через очереди сообщений (RabbitMQ, Kafka). Проектирование системы мониторинга: отслеживание качества модели (дрейф данных, дрейф концепта), технические метрики (latency, throughput), алертинг. Обеспечение объяснимости (XAI): подходы к проектированию интерфейсов для визуализации факторов влияния на решение модели. Интеграция ИИ-компонентов с существующими корпоративными системами.

Тема 6. Жизненный цикл моделей машинного обучения.

Особенности жизненного цикла ML-продукта в сравнении с традиционным ПО. Этапы: постановка бизнес-задачи и определение метрик успеха, сбор и разметка данных, исследовательский анализ данных (EDA), построение baseline-модели, итеративное улучшение модели (выбор архитектуры, настройка гиперпараметров), валидация модели. MLOps: непрерывная интеграция (CI) для кода и данных, непрерывное развёртывание (CD) моделей. Реестр моделей (model registry): версионирование, отслеживание артефактов, управление стадиями (staging, production). Развёртывание моделей: онлайн-инференс (real-time), батч-инференс (batch), модели на периферии (edge deployment). Мониторинг продакшен-моделей: детекция дрейфа, расхождение предсказаний (prediction drift), качество обслуживания (SLA). Обновление моделей: стратегии переобучения (регулярное дообучение, обучение по триггеру), A/B-тестирование, канареечный деплой (canary deployment). Утилизация и вывод из эксплуатации моделей.

Тема 7. Основы промышленной разработки ПО.

Системы контроля версий: Git (ветвление, слияние, стратегии ветвления Git Flow, GitHub Flow). Непрерывная интеграция (CI): автоматическая сборка, линтинг, статический анализ кода, выполнение тестов при каждом изменении. Инструменты CI/CD: GitHub Actions, GitLab CI, Jenkins. Контейнеризация: Docker (образы, контейнеры, Dockerfile, управление зависимостями). Оркестрация контейнеров: Kubernetes (поды, сервисы, деплойменты, масштабирование). Управление конфигурациями и секретами: переменные окружения, vault-решения. Документирование кода: docstring, автоматическая генерация документации (Sphinx, MkDocs). Работа с зависимостями: менеджеры пакетов (pip, poetry, conda), файлы requirements.txt, pyproject.toml. Логирование и трассировка: структурированное логирование (JSON), распределённая трассировка (OpenTelemetry). Практики ревью кода: цели, процесс, чек-листы, конструктивная обратная связь.

Тема 8. Обеспечение качества и безопасность программного обеспечения.

Понятие качества ПО: стандарты ISO 25010 (функциональность, надёжность, удобство использования, эффективность, сопровождаемость, переносимость). Тестирование: уровни тестирования (модульное, интеграционное, системное, приёмочное). Виды тестирования: функциональное, нагрузочное, юзабилити, регрессионное. Тест-дизайн: техники (эквивалентное разделение, анализ граничных значений, таблицы решений, попарное тестирование). Автоматизация тестирования: фреймворки (pytest, JUnit, Selenium), пирамида тестирования. Статический анализ кода: линтеры (pylint, flake8), анализаторы безопасности (bandit, SonarQube). Безопасность ПО: основ-

ные уязвимости (OWASP Top 10), безопасная разработка (SDL), управление зависимостями с учётом уязвимостей. Концепция DevSecOps: интеграция практик безопасности в CI/CD пайплайн. Управление инцидентами: мониторинг, реагирование, постмортем (blameless postmortem).

3.2. Содержание практического блока дисциплины

Очная форма обучения (полный срок)

№	Тема практического (семинарского, лабораторного) занятия
1	*2*
ПЗ 1	Реализация классов на выбранном языке программирования (Java, Python, C++) с применением принципов инкапсуляции, наследования и полиморфизма. Создание иерархий классов с переопределением методов. Применение принципов SOLID на практических примерах: рефакторинг кода с нарушенными принципами к корректной реализации. Реализация порождающих паттернов (Singleton, Factory) для управления созданием объектов. Разработка приложения с использованием поведенческих паттернов (Observer, Strategy) для организации гибкой логики. Реализация базовых структур данных: связный список, стек, очередь, бинарное дерево поиска. Решение задач на выбор оптимальной структуры данных в зависимости от требований к операциям (вставка, поиск, удаление). Анализ производительности различных реализаций.
ПЗ 2	Реализация алгоритмов сортировки (пузырьковая, быстрая, слиянием) с последующим анализом временной сложности на различных наборах данных. Построение графиков зависимости времени выполнения от размера входных данных. Реализация алгоритмов поиска (линейный, бинарный) и сравнение их эффективности. Разработка алгоритмов обхода графов (DFS, BFS) для задач поиска путей и компонент связности. Реализация алгоритма Дейкстры для поиска кратчайших путей в графах с неотрицательными весами. Решение задач динамического программирования: задача о рюкзаке, наибольшая общая подпоследовательность. Применение жадных алгоритмов к задачам выбора заявок и построения минимального остовного дерева. Вычисление асимптотической сложности разработанных решений и обоснование выбора алгоритма.
ПЗ 3	Моделирование Scrum-процесса: распределение ролей (Product Owner, Scrum Master, Development Team) в учебной группе. Формирование и приоритизация Product Backlog для выбранного проекта. Проведение Sprint Planning: оценка трудоёмкости story points, формирование Sprint Backlog. Ежедневные короткие встречи (Daily Scrum) с обсуждением прогресса и препятствий. Проведение Sprint Review: демонстрация инкремента продукта, сбор обратной связи. Проведение Sprint Retrospective: анализ прошедшего спринта, выявление улучшений. Моделирование Kanban-процесса: создание доски задач, визуализация потока, установка лимитов WIP. Сравнительный анализ эффективности методологий на основе метрик (скорость команды, время выполнения задач). Анализ кейсов применения различных методологий в реальных проектах.
ПЗ 4	Разработка устава проекта и формирование команды для учебного проекта. Проведение сбора и анализа требований: создание пользовательских историй (user stories), определение критериев приемки. Оценка трудоёмкости с использованием метода story points и планирования покером. Построение иерархической структуры работ (WBS) для выбранного проекта. Создание диаграммы Ганта с использованием инструментов (MS Project, Jira, Trello) для визуализации сроков и зависимостей. Проведение идентификации и анализа рисков: создание реестра рисков, оценка вероятности и влияния, разработка плана реагирования. Моделирование процесса управления изменениями: анализ запросов на изменение, оценка влияния на сроки и ресурсы. Расчет ключевых метрик проекта: velocity, burndown chart, анализ отклонений от плана.

ПЗ 5	Проектирование архитектуры системы для задачи классификации изображений: выделение компонентов (сервис предобработки, сервис инференса, сервис хранения результатов). Реализация REST API для взаимодействия с моделью с использованием FastAPI или Flask. Организация пайплайна обработки данных: сбор, очистка, валидация с использованием Pandas и Pydantic. Интеграция MLflow для логирования экспериментов: фиксация гиперпараметров, метрик, артефактов модели. Разработка асинхронного инференса с использованием очередей сообщений (Redis, RabbitMQ). Проектирование системы мониторинга: сбор метрик качества модели (accuracy, precision, recall), технических метрик (latency, throughput). Настройка алертинга при обнаружении дрейфа данных. Разработка интерфейса для визуализации факторов влияния на предсказание модели (SHAP, LIME).
ПЗ 6	Прохождение полного цикла ML-проекта: от постановки бизнес-задачи до развёртывания модели. Формулировка бизнес-задачи и определение метрик успеха. Загрузка и исследовательский анализ данных (EDA) с визуализацией. Построение baseline-модели с использованием простого алгоритма. Итеративное улучшение модели: выбор архитектуры, настройка гиперпараметров с использованием GridSearchCV. Версионирование данных с помощью DVC, версионирование кода с помощью Git. Регистрация модели в реестре моделей (MLflow Model Registry) с присвоением стадий. Развёртывание модели в контейнере Docker с REST API. Настройка мониторинга дрейфа данных с вычислением PSI (Population Stability Index). Проведение A/B-тестирования двух версий модели. Оформление документации проекта и презентация результатов.
ПЗ 7	Настройка репозитория Git с использованием стратегии ветвления Git Flow: создание веток feature, develop, release, hotfix. Выполнение операций слияния и разрешения конфликтов. Настройка CI/CD пайплайна в GitHub Actions или GitLab CI: автоматический запуск линтеров (pylint, flake8), статический анализ безопасности (bandit), выполнение модульных тестов при каждом push. Создание Dockerfile для приложения с оптимизацией слоёв. Сборка Docker-образа и публикация в реестре (Docker Hub). Написание манифестов для развёртывания в Kubernetes (Deployment, Service, ConfigMap). Настройка структурированного логирования с использованием библиотеки loguru или structlog. Проведение ревью кода: анализ чужого кода по чек-листу, формулирование конструктивных замечаний. Документирование кода с помощью docstring и генерация документации с Sphinx.
ПЗ 8	Написание модульных тестов с использованием pytest для тестирования функций и классов. Реализация интеграционных тестов для проверки взаимодействия компонентов. Использование фикстур и моков для изоляции тестируемого кода. Применение техник тест-дизайна: эквивалентное разделение и анализ граничных значений для разработки тестовых сценариев. Вычисление покрытия кода тестами с помощью coverage.py, анализ непокрытых участков. Проведение нагрузочного тестирования с использованием locust или JMeter: создание сценариев, анализ результатов, выявление узких мест. Статический анализ кода: настройка и запуск линтеров, интерпретация результатов. Анализ безопасности зависимостей: использование инструментов (safety, Snyk) для выявления уязвимостей в сторонних библиотеках. Проведение рефакторинга кода на основе результатов тестирования и статического анализа. Оформление отчёта о тестировании и рекомендаций по улучшению качества и безопасности.

Очно-заочная форма обучения (полный срок)

№	Тема практического (семинарского, лабораторного) занятия
1	***2***
ПЗ 5	Проектирование архитектуры системы для задачи классификации изображений: выделение компонентов (сервис предобработки, сервис инференса, сервис хранения результатов). Реализация REST API для взаимодействия с моделью с использованием

	FastAPI или Flask. Организация пайплайна обработки данных: сбор, очистка, валидация с использованием Pandas и Pydantic. Интеграция MLflow для логирования экспериментов: фиксация гиперпараметров, метрик, артефактов модели. Разработка асинхронного инференса с использованием очередей сообщений (Redis, RabbitMQ). Проектирование системы мониторинга: сбор метрик качества модели (accuracy, precision, recall), технических метрик (latency, throughput). Настройка алертинга при обнаружении дрейфа данных. Разработка интерфейса для визуализации факторов влияния на предсказание модели (SHAP, LIME).
ПЗ 6	Прохождение полного цикла ML-проекта: от постановки бизнес-задачи до развёртывания модели. Формулировка бизнес-задачи и определение метрик успеха. Загрузка и исследовательский анализ данных (EDA) с визуализацией. Построение baseline-модели с использованием простого алгоритма. Итеративное улучшение модели: выбор архитектуры, настройка гиперпараметров с использованием GridSearchCV. Версионирование данных с помощью DVC, версионирование кода с помощью Git. Регистрация модели в реестре моделей (MLflow Model Registry) с присвоением стадий. Развёртывание модели в контейнере Docker с REST API. Настройка мониторинга дрейфа данных с вычислением PSI (Population Stability Index). Проведение A/B-тестирования двух версий модели. Оформление документации проекта и презентация результатов.
ПЗ 7	Настройка репозитория Git с использованием стратегии ветвления Git Flow: создание веток feature, develop, release, hotfix. Выполнение операций слияния и разрешения конфликтов. Настройка CI/CD пайплайна в GitHub Actions или GitLab CI: автоматический запуск линтеров (pylint, flake8), статический анализ безопасности (bandit), выполнение модульных тестов при каждом push. Создание Dockerfile для приложения с оптимизацией слоёв. Сборка Docker-образа и публикация в реестре (Docker Hub). Написание манифестов для развёртывания в Kubernetes (Deployment, Service, ConfigMap). Настройка структурированного логирования с использованием библиотеки loguru или structlog. Проведение ревью кода: анализ чужого кода по чек-листу, формулирование конструктивных замечаний. Документирование кода с помощью docstring и генерация документации с Sphinx.
ПЗ 8	Написание модульных тестов с использованием pytest для тестирования функций и классов. Реализация интеграционных тестов для проверки взаимодействия компонентов. Использование фикстур и моков для изоляции тестируемого кода. Применение техник тест-дизайна: эквивалентное разделение и анализ граничных значений для разработки тестовых сценариев. Вычисление покрытия кода тестами с помощью coverage.py, анализ непокрытых участков. Проведение нагрузочного тестирования с использованием locust или JMeter: создание сценариев, анализ результатов, выявление узких мест. Статический анализ кода: настройка и запуск линтеров, интерпретация результатов. Анализ безопасности зависимостей: использование инструментов (safety, Snyk) для выявления уязвимостей в сторонних библиотеках. Проведение рефакторинга кода на основе результатов тестирования и статического анализа. Оформление отчёта о тестировании и рекомендаций по улучшению качества и безопасности.

3.3. Образовательные технологии
Очная форма обучения (полный срок)

№	Тема занятия	Вид учебного занятия	Форма / Методы интерактивного обучения	% учебного времени
1	*2*	*3*	*4*	*5*
1	Принципы ООП. Объектное проектирование. Структуры данных.	ПЗ	Работа в парах, Мозговой штурм, Дискуссионные технологии, Интерактивная визуали-	25

			зация (моделирование объектов), Кейс-стади (разбор объектных моделей)	
2	Алгоритмы и оценка их сложности	ПЗ	Семинар-дискуссия, Работа в группах с презентацией, Кейс-стади, Ролевая игра (архитектор/разработчик), Проектно-ориентированное обучение	25
3	Методологии программной инженерии	ПЗ	Групповое решение задач, Интерактивная визуализация алгоритмов, Мозговой штурм, Работа в парах, Практикум с кодированием	25
4	Управление проектами разработки программного обеспечения	ПЗ	Дискуссионные технологии, Интерактивная визуализация (анализ работы алгоритмов), Конкурс (на лучшее решение задачи), Групповое решение задач, Кейс-стади	25
5	Проектирование ПО для решения задач в области ИИ	ПЗ	Ролевая игра (Scrum-команда), Проектно-ориентированное обучение, Семинар-дискуссия, Работа в группах, Кейс-стади (разбор реальных проектов)	25
6	Жизненный цикл моделей машинного обучения	ПЗ	Практикум в группах, Взаимооценка (peer review), Кейс-стади (анализ багов), Дискуссионные технологии, Интерактивная визуализация процессов тестирования	25
7	Основы промышленной разработки ПО	ПЗ	Проектно-ориентированное обучение, Работа в группах с презентацией, Мозговой штурм, Кейс-стади (разбор моделей ИИ), Дискуссия по этическим вопросам	25
8	Обеспечение качества и безопасность программного обеспечения	ПЗ	Работа в командах, Ролевая игра (DevOps-команда), Проектно-ориентированное обучение, Кейс-стади (реальные IT-проекты), Взаимооценка и code review	25
Итого				**25%**

Очно-заочная форма обучения (полный срок)

№	**Тема занятия**	**Вид учебного занятия**	Форма / Методы интерактивного обучения	**% учебного времени**
1	*2*	*3*	*4*	*5*
5	Проектирование ПО для решения задач в области ИИ	ПЗ	Ролевая игра (Scrum-команда), Проектно-ориентированное обучение, Семинар-дискуссия, Работа в группах, Кейс-стади	25

			(разбор реальных проектов)	
6	Жизненный цикл моделей машинного обучения	ПЗ	Практикум в группах, Взаимооценка (peer review), Кейс-стади (анализ багов), Дискуссионные технологии, Интерактивная визуализация процессов тестирования	25
7	Основы промышленной разработки ПО	ПЗ	Проектно-ориентированное обучение, Работа в группах с презентацией, Мозговой штурм, Кейс-стади (разбор моделей ИИ), Дискуссия по этическим вопросам	25
8	Обеспечение качества и безопасность программного обеспечения	ПЗ	Работа в командах, Ролевая игра (DevOps-команда), Проектно-ориентированное обучение, Кейс-стади (реальные IT-проекты), Взаимооценка и code review	25
Итого				**25%**

Раздел 4. Организация самостоятельной работы обучающихся

4.1. Организация самостоятельной работы обучающихся

№	Тема дисциплины	№ вопросов	№ рекомендуемой литературы
1	*2*	*3*	*4*
1	Принципы ООП. Объектное проектирование. Структуры данных.	1-5	1, 2, 9, 10, 11, 14
2	Алгоритмы и оценка их сложности	6-10	1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 13, 14, 15
3	Методологии программной инженерии	11-15	3, 4, 5, 9, 10, 11, 12, 13, 14, 15
4	Управление проектами разработки программного обеспечения	16-20	1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 13, 14, 15
5	Проектирование ПО для решения задач в области ИИ	21-25	3, 4, 5, 9, 10, 11, 12, 13, 14, 15
6	Жизненный цикл моделей машинного обучения	26-30	1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 13, 14, 15
7	Основы промышленной разработки ПО	31-35	3, 4, 5, 9, 10, 11, 12, 13, 14, 15
8	Обеспечение качества и безопасность программного обеспечения	36-40	3, 4, 5, 9, 10, 11, 12, 13, 14, 15

Перечень вопросов, выносимых на самостоятельную работу обучающихся

1. Что такое программная инженерия и каковы её основные цели?
2. В чем заключаются основные принципы объектно-ориентированного программирования?
3. Что такое класс и объект в ООП?
4. В чем разница между инкапсуляцией и абстракцией?
5. Как реализуется наследование и какие его виды существуют?
6. Что такое полиморфизм и какие формы он принимает?
7. Что такое UML и для чего он используется?
8. Какие типы UML-диаграмм вы знаете?
9. В чем заключаются принципы SOLID?
10. Что такое паттерны проектирования и зачем они нужны?
11. Приведите примеры порождающих паттернов.
12. Приведите примеры структурных паттернов.
13. Приведите примеры поведенческих паттернов.
14. Что такое структура данных и какие виды структур данных существуют?
15. Чем отличается стек от очереди?
16. Что такое дерево и какие виды деревьев применяются в программировании?
17. Что такое граф и где он используется?
18. Что такое хеш-таблица и как она работает?
19. Что такое алгоритм и какими свойствами он обладает?
20. Что такое временная и пространственная сложность алгоритма?
21. Что означает нотация Big O?
22. Какие алгоритмы сортировки вы знаете и в чем их различия?

23. Какие алгоритмы поиска существуют?
24. Что такое жизненный цикл разработки программного обеспечения (SDLC)?
25. Какие модели разработки ПО вы знаете?
26. В чем особенности каскадной модели разработки?
27. Что такое Agile и в чем его преимущества?
28. В чем различия между Scrum и Kanban?
29. Что такое требования к программному обеспечению?
30. Какие методы сбора и анализа требований существуют?
31. Что такое пользовательские истории (user stories)?
32. Что такое тестирование программного обеспечения?
33. Какие виды тестирования существуют?
34. В чем различие между модульным и интеграционным тестированием?
35. Что такое автоматизированное тестирование?
36. Что такое архитектура программного обеспечения?
37. В чем различие между монолитной и микросервисной архитектурой?
38. Что такое CI/CD и зачем он используется?
39. Какие особенности имеет проектирование программных систем в области искусственного интеллекта?
40. Какие этические проблемы возникают при разработке ИИ-систем?

4.2. Перечень учебно-методического обеспечения самостоятельной работы обучающихся

Самостоятельная работа обучающихся обеспечивается следующими учебно-методическими материалами:

1. Указаниями в рабочей программе по дисциплине (п.4.1.)
2. Лекционные материалы в составе учебно-методического комплекса по дисциплине
3. Заданиями и методическими рекомендациями по организации самостоятельной работы обучающихся в составе учебно-методического комплекса по дисциплине.
4. Глоссарием по дисциплине в составе учебно-методического комплекса по дисциплине.

Раздел 5. Фонд оценочных средств для проведения текущего контроля и промежуточной аттестации обучающихся

Фонд оценочных средств по дисциплине представляет собой совокупность контролирующих материалов, предназначенных для измерения уровня достижения обучающимися установленных результатов образовательной программы. ФОС по дисциплине используется при проведении оперативного контроля и промежуточной аттестации обучающихся. Требования к структуре и содержанию ФОС дисциплины регламентируются Положением о фонде оценочных материалов по программам высшего образования – программам бакалавриата, магистратуры.

5.1. Паспорт фонда оценочных средств

Очная форма обучения (полный срок)

№	Контролируемые разделы (темы) дисциплины	Оценочные средства			
		Л	ПЗ (ЛЗ, СЗ)	СРО	Код индикатора и дескриптора достижения компетенций
1	*2*	*3*	*4*	*5*	*6*
1	Принципы ООП. Объектное проектирование. Структуры данных.	УО	ЗЗ, МШ	ПРВ	ИД-1 ОПК- 6.1 ИД-2 ОПК- 6.2
2	Алгоритмы и оценка их сложности	УО	ЗЗ, Д	ПРВ	ИД-3 ОПК- 6.1 ИД-4 ОПК- 6.2
3	Методологии программной инженерии	УО	ЗЗ, Д, МШ	ПРВ	ИД-5 ОПК- 6.1 ИД-6 ОПК- 6.2
4	Управление проектами разработки программного обеспечения	УО	ЗЗ, Д, МП	ПРВ	ИД-1 ПК- 2.1 ИД-2 ОПК- 2.2
5	Проектирование ПО для решения задач в области ИИ	УО	ЗЗ, МШ	ПРВ	ИД-3 ПК- 2.1 ИД-4 ПК- 2.2
6	Жизненный цикл моделей машинного обучения	УО	ЗЗ, МШ	ПРВ	ИД-3 ПК- 2.1 ИД-4 ОПК- 2.2
7	Основы промышленной разработки ПО	УО	ЗЗ, МШ	ПРВ	ИД-5 ПК- 2.1 ИД-6 ОПК- 2.2
8	Обеспечение качества и безопасность программного обеспечения	УО	ЗЗ, МШ	ПРВ	ИД-5 ПК- 2.1 ИД-6 ПК- 2.2

Очно-заочная форма обучения (полный срок)

№	Контролируемые разделы (темы) дисциплины	Оценочные средства			
		Л	ПЗ (ЛЗ, СЗ)	СРО	Код индикатора и дескриптора достижения компетенций
1	*2*	*3*	*4*	*5*	*6*
1	Принципы ООП. Объектное проектирование. Структуры данных.	УО		ПРВ	ИД-1 ОПК- 6.1 ИД-2 ОПК- 6.2
2	Алгоритмы и оценка их сложности	УО		ПРВ	ИД-3 ОПК- 6.1 ИД-4 ОПК- 6.2
3	Методологии программной инженерии	УО		ПРВ	ИД-5 ОПК- 6.1 ИД-6 ОПК- 6.2
4	Управление проектами разработки программного обеспечения	УО		ПРВ	ИД-1 ПК- 2.1 ИД-2 ОПК- 2.2

5	Проектирование ПО для решения задач в области ИИ		ЗЗ, МШ	ПРВ	ИД-3 ПК- 2.1 ИД-4 ПК- 2.2
6	Жизненный цикл моделей машинного обучения		ЗЗ, МШ	ПРВ	ИД-3 ПК- 2.1 ИД-4 ОПК- 2.2
7	Основы промышленной разработки ПО		ЗЗ, МШ	ПРВ	ИД-5 ПК- 2.1 ИД-6 ОПК- 2.2
8	Обеспечение качества и безопасность программного обеспечения		ЗЗ, МШ	ПРВ	ИД-5 ПК- 2.1 ИД-6 ПК- 2.2

Условные обозначения оценочных средств (Столбцы 3, 4, 5):

ЗЗ – защита выполненных заданий (творческих, расчетных и т.д.), представление презентаций;

ПРВ – проверка рефератов, отчетов, рецензий, аннотаций, конспектов, графического материала, эссе, переводов, решений заданий, выполненных заданий в электронном виде и т.д.;

МШ – Метод мозгового штурма;

Д – Дискуссия, полемика, диспут, дебаты;

МП – Метод проектов.

5.2. Тематика письменных работ обучающихся

1. Роль программной инженерии в современном мире информационных технологий
2. Основные принципы объектно-ориентированного программирования и их применение
3. Сравнительный анализ объектно-ориентированного и процедурного подходов
4. Применение принципов SOLID при разработке программного обеспечения
5. Использование UML-диаграмм в процессе проектирования программных систем
6. Паттерны проектирования: классификация и практическое значение
7. Выбор структур данных в зависимости от поставленной задачи
8. Анализ эффективности алгоритмов и оценка их сложности
9. Сравнение алгоритмов сортировки и их применение на практике
10. Жизненный цикл разработки программного обеспечения: этапы и модели
11. Сравнительный анализ каскадной и гибкой методологий разработки
12. Методологии Agile: принципы, преимущества и недостатки
13. Управление требованиями в процессе разработки программного обеспечения
14. Методы тестирования программного обеспечения и их эффективность
15. Автоматизация тестирования: инструменты и подходы
16. Архитектурные подходы в разработке ПО: монолит vs микросервисы
17. Практика использования систем контроля версий в командной разработке
18. Непрерывная интеграция и доставка (CI/CD) в современном DevOps
19. Особенности проектирования программных систем в области искусственного интеллекта
20. Этические и социальные аспекты разработки ИИ-систем

5.3. Перечень вопросов промежуточной аттестации по дисциплине

Вопросы к экзамену:

1. Основные принципы объектно-ориентированного программирования (ООП) и их значение
2. Классы, объекты и их взаимодействие в программных системах
3. Принципы объектного проектирования и их применение в разработке ПО
4. Основные паттерны проектирования и их классификация
5. Базовые структуры данных и их применение в программировании
6. Сравнение линейных и нелинейных структур данных

7. Жизненный цикл программного обеспечения (SDLC) и его этапы
8. Гибкие методологии разработки ПО (Agile, Scrum, Kanban)
9. Архитектура программного обеспечения: основные понятия и виды
10. Микросервисная и монолитная архитектуры: сравнительный анализ
11. Управление требованиями в процессе разработки программного обеспечения
12. Методы тестирования программного обеспечения и их классификация
13. Основы алгоритмизации и оценка сложности алгоритмов (Big O)
14. Основные алгоритмы сортировки и поиска: сравнительный анализ
15. Принципы проектирования пользовательских интерфейсов в программных системах
16. Роль человеко-машинного взаимодействия (HCI) в разработке ПО
17. Проектирование программных систем для задач искусственного интеллекта
18. Основные этапы жизненного цикла моделей машинного обучения (MLOps)
19. Интеграция моделей ИИ в программные системы и их развертывание
20. Основы промышленной разработки ПО: CI/CD, Git и DevOps-подход

Раздел 6. Оценочные средства промежуточной аттестации (с ключами)

1. Что является основным принципом ООП?
А) Последовательное выполнение команд
Б) Инкапсуляция данных и поведения
В) Использование только функций
Г) Отсутствие структур
Ответ: Б

2. Что такое полиморфизм?
А) Создание новых классов
Б) Сокрытие данных
В) Возможность объектов разных классов использовать единый интерфейс
Г) Удаление объектов
Ответ: В

3. Какая структура данных работает по принципу LIFO?
А) Очередь
Б) Стек
В) Граф
Г) Хеш-таблица
Ответ: Б

4. Какой алгоритм имеет среднюю сложность O(n log n)?
А) Bubble Sort
Б) Quick Sort
В) Linear Search
Г) Selection Sort
Ответ: Б

5. Что означает Agile?
А) Жёсткую последовательную разработку
Б) Гибкую методологию разработки ПО
В) Язык программирования
Г) Тип базы данных
Ответ: Б

6. Что такое SDLC?

А) Язык программирования
Б) Жизненный цикл разработки ПО
В) Тип алгоритма
Г) Система хранения данных
Ответ: Б

7. Что такое UML?
А) Язык программирования
Б) Метод тестирования
В) Язык визуального моделирования систем
Г) База данных
Ответ: В

8. Что такое CI/CD?
А) Метод сортировки
Б) Процесс непрерывной интеграции и доставки
В) Алгоритм поиска
Г) Тип структуры данных
Ответ: Б

9. Что характерно для микросервисной архитектуры?
А) Один большой модуль
Б) Независимые сервисы
В) Отсутствие сети
Г) Только локальное выполнение
Ответ: Б

10. Что используется для оценки качества модели ИИ?
А) FPS и RAM
Б) Accuracy и F1-score
В) Скорость процессора
Г) Размер файла
Ответ: Б

Раздел 7. Перечень учебной литературы, необходимой для освоения дисциплины

7.1. Основная литература

1. Фаулер М., Рефакторинг: улучшение существующего кода, 2019
2. Буч Г., Объектно-ориентированный анализ и проектирование с примерами приложений, 2017
3. Брукс Ф., Мифический человеко-месяц, 2018
4. Басс Л., Клементс П., Кацман Р., Архитектура программного обеспечения: практика и принципы, 2020
5. Соммервилл И., Инженерия программного обеспечения, 2021

7.2. Дополнительная литература

1. Gamma E., Helm R., Johnson R., Vlissides J., *Design Patterns: Elements of Reusable Object-Oriented Software*, 2015
2. Martin R., *Clean Architecture*, 2018
3. Martin R., *Clean Code*, 2008
4. Шоу М., *Software Architecture: Foundations, Theory, and Practice*, 2019

5. Russel S., Norvig P., *Artificial Intelligence: A Modern Approach*, 2020

7.3. Перечень ресурсов информационно-телекоммуникационной сети «Интернет»

1. MDN Web Docs, *JavaScript and Web Development Reference*, 2024
2. Refactoring Guru, *Design Patterns and Principles*, 2023
3. GeeksforGeeks, *Data Structures and Algorithms Tutorials*, 2024
4. GitHub Docs, *Version Control with Git*, 2025
5. Microsoft Learn, *DevOps and CI/CD Pipelines*, 2025

Раздел 8. Материально-техническая база и информационные технологии

Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине:

Материально-техническое обеспечение дисциплины **«Программная инженерия»** включает в себя учебные аудитории для проведения занятий, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, самостоятельной работы обучающихся.

Учебные аудитории укомплектованы специализированной мебелью и техническими средствами обучения. Помещения для самостоятельной работы обучающихся оснащены компьютерной техникой с возможностью подключения к сети Интернет.

Дисциплина может реализовываться с применением дистанционных технологий обучения. Специфика реализации дисциплины с применением дистанционных технологий обучения устанавливается дополнением к рабочей программе. В части не противоречащей специфике, изложенной в дополнении к программе, применяется настоящая рабочая программа.

Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине с применением дистанционных образовательных технологий включает в себя:

Компьютерная техника, расположенная в учебном корпусе Института (ул. Качинцев, 63, кабинет Центра дистанционного обучения):

1. Intel i 3 3.4Ghz\ОЗУ 4Gb\500GB\RadeonHD5450
2. Intel PENTIUM 2.9GHz\ОЗУ 4GB\500GB

3 личные электронные устройства (компьютеры, ноутбуки, планшеты и иное), а также средства связи преподавателей и студентов.

Информационные технологии, необходимые для осуществления образовательного процесса по дисциплине с применением дистанционных образовательных технологий включают в себя:

- система дистанционного обучения (СДО) (Learning Management System) (LMS) Moodle (Modular Object-Oriented Dynamic Learning Environment);
- электронная почта;
- система компьютерного тестирования;
- Цифровой образовательный ресурс IPR SMART;
- система интернет-связи skype;
- телефонная связь;
- ПО для организации конференций.

Обучение обучающихся инвалидов и обучающихся с ограниченными возможностями здоровья осуществляется посредством применения специальных технических средств в зависимости от вида нозологии.

При проведении учебных занятий по дисциплине используются мультимедийные комплексы, электронные учебники и учебные пособия, адаптированные к ограничениям здоровья обучающихся.

Лекционные аудитории оборудованы мультимедийными кафедрами, подключенными к звуковым колонкам, позволяющими усилить звук для категории слабослышащих обучающихся, а также проекционными экранами, которые увеличивают изображение в несколько раз и позволяют воспринимать учебную информацию обучающимся с нарушениями зрения.

При обучении лиц с нарушениями слуха используется усилитель слуха для слабослышащих людей Cyber Ear модель HAP-40, помогающий обучаемым лучше воспринимать учебную информацию.

Обучающиеся с ограниченными возможностями здоровья, обеспечены печатными и электронными образовательными ресурсами (программы, учебники, учебные пособия, материалы для самостоятельной работы и т.д.) в формах, адаптированных к ограничениям их здоровья и восприятия информации:

для лиц с нарушениями зрения:

- в форме электронного документа;
- в форме аудиофайла;

для лиц с нарушениями слуха:

- в печатной форме;
- в форме электронного документа;

для лиц с нарушениями опорно-двигательного аппарата:

- в печатной форме;
- в форме электронного документа;
- в форме аудиофайла.

Раздел 9. Методические указания для обучающихся по освоению дисциплины

Дисциплина включает практические занятия, самостоятельную работу обучающегося.

В ходе изучения дисциплины «Программная инженерия» перед обучающимися стоит задача не только закрепить знания о сложных информационных явлениях, о чем свидетельствует содержание тематического плана, глубоко разобраться в объемном учебном материале, но и сформировать у себя на основе полученных компьютерных знаний соответствующие профессионально важные качества.

Практические занятия – один из самых эффективных видов учебных занятий, на которых обучающиеся учатся творчески работать с различной информацией, являются также действенной формой активизации самостоятельной работы обучающихся.

Целью практических занятий является закрепление полученных в ходе лекций, а также в ходе самостоятельной работы над учебной и специальной литературой, знаний, умений и навыков. На практических занятиях особо обращается внимание на умение обучающихся проявлять элементы творчества в процессе самостоятельной работы, применять полученные знания на практике.

Практические занятия занимают центральное место в учебном процессе, так как позволяют на завершающем этапе усвоения материала, после прослушанной лекции и самостоятельного поиска дополнительных сведений по рассматриваемой проблематике, окончательно уточнить, сформировать свои позиции в ходе работы в составе учебной группы.

Основное в подготовке и проведении практикума – это самостоятельная работа обучающегося над изучением темы лекционного материала. Практические занятия проводятся по специальным планам – заданиям, которые содержатся в материалах, подготовленных на кафедре. Обучающийся обязан точно знать план занятия либо конкретное задание к нему.

При подготовке к практическим занятиям следует чаще обращаться к справочной литературе, полнее использовать консультации (групповые и индивидуальные, устные и письменные) с преподавателями, которые читают лекции и проводят практикумы.

Таким образом, в процессе подготовке к практическому занятию рекомендуется:

- ознакомиться с вопросами плана;
- прочитать конспект лекции по изучаемой теме;
- прочитать соответствующие главы учебников, статьи;
- просмотреть перечень научных источников, предлагаемых в рабочей программе, выбрав несколько из них для углубленного изучения данной темы.

По каждому практическому заданию обучающиеся отчитываются преподавателю, оформляя письменный отчёт, в котором сохраняют результаты своей работы в виде файлов. Результаты выполнения практических заданий оцениваются с учетом теоретических знаний по соответствующим вопросам дисциплины и уровнем владения практическими навыками при работе на компьютере.

Для углубленного изучения и освоения материала целесообразно выполнение практических работ, наряду с другими различными формами обучения обучающихся: тесты, задачи, упражнения, которые используются при проведении практических занятий, выполнении контрольных и аудиторных работ, а также при самостоятельном изучении данной дисциплины.

Одним из наиболее интенсивных способов изучения дисциплины является самостоятельное выполнение практических работ, на которых вырабатываются навыки по дисциплине «Программная инженерия».

СРО позволяет глубже освоить теоретические и практические вопросы, понять принципы дисциплины «Программная инженерия».

Основными задачами организации процесса самостоятельной работы по дисциплине являются:

- приобретение знаний по теоретическим основам дисциплины «Программная инженерия», являющихся дополнением к материалу лекционных аудиторных занятий;
- приобретение практических навыков по дисциплине «Программная инженерия».

Основные формы реализации СРО – изучение учебно-методической литературы по дисциплине «Программная инженерия». В качестве базовой литературы можно использовать учебники и учебные пособия, согласно приведенному списку в разделе 6 рабочей программы, а также лю-

бые другие источники информации, такие как электронные учебники, обучающие и энциклопедические сайты, публикации журналов и конференций.

Обучающийся допускается к зачетному занятию по результатам успешного выполнения всех практических заданий и самостоятельной работы.

Учебно-методическое издание

Рабочая программа учебной дисциплины

Программная инженерия

(Наименование дисциплины в соответствии с учебным планом)

Скоробогатченко Дмитрий Анатольевич

(Фамилия, Имя, Отчество составителя)