

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего
образования
"САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ"

Кафедра № 32

УТВЕРЖДАЮ

Руководитель образовательной программы

доц., к.т.н., доц.

(должность, уч. степень, звание)

О.Я. Солёная

 (инициалы, фамилия)

(подпись)

«18» февраля 2026 г

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

«Объектно-ориентированное программирование»

(Наименование дисциплины)

Код направления подготовки/ специальности	13.03.02
Наименование направления подготовки/ специальности	Электроэнергетика и электротехника
Наименование направленности/ специализации	Цифровая энергетика
Форма обучения	очно-заочная
Год приема	2026

Санкт-Петербург– 2026

Лист согласования рабочей программы дисциплины

Программу составил (а)

доц., к.т.н.

(должность, уч. степень, звание)



18.02.2026

(подпись, дата)

Н.В. Савельев

(инициалы, фамилия)

Программа одобрена на заседании кафедры № 32

«18» февраля 2026 г, протокол № 8

Заведующий кафедрой № 32

к.т.н., доц.

(уч. степень, звание)



18.02.2026

(подпись, дата)

С.В. Солёный

(инициалы, фамилия)

Заместитель директора института №3 по методической работе

доц., к.т.н.

(должность, уч. степень, звание)



18.02.2026

(подпись, дата)

Н.В. Решетникова

(инициалы, фамилия)

Аннотация

Дисциплина «Объектно-ориентированное программирование» входит в образовательную программу высшего образования – программу бакалавриата по направлению подготовки/ специальности 13.03.02 «Электроэнергетика и электротехника» направленности/специализации «Цифровая энергетика». Дисциплина реализуется кафедрой «№32».

Дисциплина не является обязательной при освоении обучающимся образовательной программы и направлена на углубленное формирование следующих компетенций:

ПК-2 «Способен участвовать в научных исследованиях объектов профессиональной деятельности»

Содержание дисциплины охватывает круг вопросов, связанных с объектно-ориентированным программированием на высокоуровневых языках.

Преподавание дисциплины предусматривает следующие формы организации учебного процесса: лабораторные работы, самостоятельная работа обучающегося.

Программой дисциплины предусмотрены следующие виды контроля: текущий контроль успеваемости, промежуточная аттестация в форме дифференцированного зачета (6 семестр).

Общая трудоемкость освоения дисциплины составляет 2 зачетных единицы, 72 часа.

Язык обучения по дисциплине «русский»

1. Перечень планируемых результатов обучения по дисциплине

1.1. Цели преподавания дисциплины

Целью дисциплины является формирование у студентов знаний, умений и навыков в области объектно-ориентированного программирования на высокоуровневых языках.

1.2. Дисциплина является факультативной дисциплиной по направлению образовательной программы высшего образования (далее – ОП ВО).

1.3. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения ОП ВО.

В результате изучения дисциплины обучающийся должен обладать следующими компетенциями или их частями. Компетенции и индикаторы их достижения приведены в таблице 1.

Таблица 1 – Перечень компетенций и индикаторов их достижения

Категория (группа) компетенции	Код и наименование компетенции	Код и наименование индикатора достижения компетенции
Профессиональные компетенции	ПК-2 Способен участвовать в научных исследованиях объектов профессиональной деятельности	ПК-2.Д.4 использует соответствующее программное обеспечение для оформления результатов научно-исследовательских работ

2. Место дисциплины в структуре ОП

Дисциплина может базироваться на знаниях, ранее приобретенных обучающимися при изучении следующих дисциплин:

- «Математика. Математический анализ»,
- «Информатика»,
- «Алгоритмизация и программирование».

Знания, полученные при изучении материала данной дисциплины, имеют как самостоятельное значение, так и могут использоваться при изучении других дисциплин:

- «Программирование микроконтроллеров»,
- «Системы и методы искусственного интеллекта в электроэнергетике»,
- «Математические методы исследований».

3. Объем и трудоемкость дисциплины

Данные об общем объеме дисциплины, трудоемкости отдельных видов учебной работы по дисциплине (и распределение этой трудоемкости по семестрам) представлены в таблице 2.

Таблица 2 – Объем и трудоемкость дисциплины

Вид учебной работы	Всего	Трудоемкость по семестрам
		№6
1	2	3
Общая трудоемкость дисциплины, ЗЕ/ (час)	2/ 72	2/ 72
Из них часов практической подготовки	17	17
Аудиторные занятия, всего час.	34	34
в том числе:		
лекции (Л), (час)	17	17
практические/семинарские занятия (ПЗ),		

(час)		
лабораторные работы (ЛР), (час)	17	17
курсовой проект (работа) (КП, КР), (час)		
экзамен, (час)		
Самостоятельная работа , всего (час)	38	38
Вид промежуточной аттестации: зачет, дифф. зачет, экзамен (Зачет, Дифф. зач, Экз.)	Дифф. зач.	Дифф. зач.

4. Содержание дисциплины

4.1. Распределение трудоемкости дисциплины по разделам и видам занятий.

Разделы, темы дисциплины и их трудоемкость приведены в таблице 3.

Таблица 3 – Разделы, темы дисциплины, их трудоемкость

Разделы, темы дисциплины	Лекции (час)	ПЗ (СЗ) (час)	ЛР (час)	КП/КР (час)	СР (час)
Семестр 6					
Раздел 1. Введение в объектно-ориентированное программирование Тема 1.1. Основные концепции ООП: инкапсуляция, наследование, полиморфизм, абстракция. Тема 1.2. Сравнение парадигм программирования: процедурное программирование, функциональное программирование, объектно-ориентированное программирование. Тема 1.3. Преимущества и недостатки ООП.	1				1
Раздел 2. Классы и объекты Тема 2.1. Класс как основная единица ООП: определение класса, поля (атрибуты) и методы класса. Тема 2.2. Объект как экземпляр класса: создание объектов, инициализация объектов. Тема 2.3. Модификаторы доступа: Public, private, protected, геттеры и сеттеры. Тема 2.4. Конструкторы и деструкторы: перегрузка конструкторов, управление жизненным циклом объекта.	2		2		4
Раздел 3. Инкапсуляция Тема 3.1. Скрытие данных: принципы инкапсуляции, разделение интерфейса и реализации. Тема 3.2. Работа с приватными и защищенными членами класса: методы доступа, реализация геттеров и сеттеров.	1		2		4

<p>Раздел 4. Наследование</p> <p>Тема 4.1. Основы наследования: базовый и производный класс, расширение функциональности.</p> <p>Тема 4.2. Типы наследования: одиночное наследование, множественное наследование.</p> <p>Тема 4.3. Переопределение методов: переопределение в производных классах, использование ключевого слова <code>super</code> или его аналогов.</p> <p>Тема 4.4. Абстрактные классы и интерфейсы: частичная реализация классов, полиморфизм через интерфейсы.</p>	1		2		4
<p>Раздел 5. Полиморфизм</p> <p>Тема 5.1. Статический и динамический полиморфизм: перегрузка методов, переопределение методов.</p> <p>Тема 5.2. Виртуальные функции: позднее связывание, использование абстрактных методов.</p> <p>Тема 5.3. Примеры применения полиморфизма: работа с коллекциями, объектов, реализация шаблонов проектирования.</p>	2		2		4
<p>Раздел 6. Абстракция</p> <p>Тема 6.1. Абстрактные классы: определение и использование, ограничения при работе с абстрактными классами.</p> <p>Тема 6.2. Интерфейсы: определение контрактов, реализация нескольких интерфейсов.</p> <p>Тема 6.3. Примеры использования абстракции: проектирование сложных систем, упрощение взаимодействия между компонентами.</p>	2		2		4
<p>Раздел 7. Обработка исключений</p> <p>Тема 7.1. Основы обработки ошибок: исключения как механизм управления ошибками, иерархия исключений.</p> <p>Тема 7.2. Блоки <code>try-catch-finally</code>: ловля и обработка исключений, освобождение ресурсов.</p> <p>Тема 7.3. Создание пользовательских исключений.</p>	2		2		4
<p>Раздел 8. Работа с коллекциями</p> <p>Тема 8.1. Стандартные коллекции: списки, множества, словари, особенности работы с коллекциями в ООП.</p> <p>Тема 8.2. Итераторы и генераторы: итерирование по коллекциям, создание собственных итераторов.</p> <p>Тема 8.3. Шаблоны проектирования для работы с коллекциями.</p>	2		2		4

Раздел 9. Шаблоны проектирования Тема 9.1. Основные паттерны проектирования: порождающие паттерны (Singleton, Factory, Builder), структурные паттерны (Adapter, Decorator, Composite), поведенческие паттерны (Observer, Strategy, Command). Тема 9.2. Примеры реализации паттернов: практическое применение в реальных проектах.	2		3		4
Раздел 10. Тестирование объектно-ориентированного кода Тема 10.1. Юнит-тестирование: написание тестов для классов и методов, использование фреймворков для тестирования. Тема 10.2. Моки и стабы: тестирование зависимостей, изоляция тестируемого кода.	1				3
Раздел 11. Современные подходы к ООП Тема 11.1. Фреймворки и библиотеки: примеры использования ООП в популярных фреймворках (например, Java Spring, .NET, Python Django). Тема 11.2. ООП в многопоточном программировании: синхронизация потоков, безопасность данных. Тема 11.3. ООП и функциональное программирование: комбинирование парадигм, примеры гибридных подходов.	1				2
Итого в семестре:	17		17		38
Итого	17	0	17	0	38

Практическая подготовка заключается в непосредственном выполнении обучающимися определенных трудовых функций, связанных с будущей профессиональной деятельностью.

4.2. Содержание разделов и тем лекционных занятий.

Содержание разделов и тем лекционных занятий приведено в таблице 4.

Таблица 4 – Содержание разделов и тем лекционного цикла

Номер раздела	Название и содержание разделов и тем лекционных занятий
1.	Раздел 1. Введение в объектно-ориентированное программирование Тема 1.1. Основные концепции ООП: инкапсуляция, наследование, полиморфизм, абстракция. Лекция-беседа. Понимать сущность и назначение четырёх базовых принципов ООП, уметь идентифицировать проявления каждого принципа в коде и предметной области, формировать ментальную модель объекта как единицы состояния и поведения. Объект, класс, состояние, поведение, интерфейс. Инкапсуляция: сокрытие реализации, защита инвариантов. Наследование: иерархии, переопределение, специализация. Полиморфизм: единый интерфейс, множественная реализация. Абстракция: выделение существенного, игнорирование деталей Тема 1.2. Сравнение парадигм программирования: процедурное программирование, функциональное программирование, объектно-ориентированное

	<p>программирование. Лекция-беседа. Различать фундаментальные подходы к организации кода в трёх парадигмах, анализировать задачи и выбирать адекватную парадигму или их комбинацию, понимать исторический контекст и эволюцию парадигм. Процедурная парадигма: функции, последовательность, глобальное состояние. Функциональная парадигма: чистые функции, неизменяемость, композиция. Объектно-ориентированная парадигма: объекты, сообщения, инкапсуляция. Мультипарадигменность: гибридные языки и стили.</p> <p>Тема 1.3. Преимущества и недостатки ООП. Лекция-беседа. Критически оценивать применимость ООП в различных контекстах, понимать компромиссы между гибкостью, производительностью и сложностью, формировать сбалансированное представление об ООП как об инструменте, а не догме. Преимущества: модульность, повторное использование, масштабируемость, выразительность. Недостатки: избыточная абстракция, сложность отладки, накладные расходы, кривая обучения. Контекстуальная адекватность: когда ООП уместно, а когда — нет. Альтернативы и дополнения: процедурный подход, ФП, реактивное программирование.</p>
2.	<p>Раздел 2. Классы и объекты</p> <p>Тема 2.1. Класс как основная единица ООП: определение класса, поля (атрибуты) и методы класса. Лекция-беседа. Сформировать чёткое представление о классе как абстрактной модели сущности, научиться проектировать структуру класса: отделять состояние (данные) от поведения (методы), понимать роль класса как «чертежа» для создания объектов. Класс, сущность предметной области, модель данных. Поля (атрибуты): тип данных, имя, состояние объекта. Методы: сигнатура, параметры, возвращаемое значение, область видимости. Статические и динамические члены класса (вводное знакомство). Соглашения об именовании и документировании классов.</p> <p>Тема 2.2. Объект как экземпляр класса: создание объектов, инициализация объектов. Лекция-беседа. Освоить механизм инстанцирования объектов и работу со ссылками, понимать различие между значением и ссылкой, идентичностью и равенством, научиться управлять состоянием объектов после создания. Объект, экземпляр, инстанцирование. Оператор создания объекта (new, __init__, инициализаторы). Ссылка на объект, null/None/nil, разыменование. Куча (heap) vs стек (stack): базовая модель размещения. Сравнение объектов: ссылочное равенство (==) vs содержательное равенство (equals(), __eq__).</p> <p>Тема 2.3. Модификаторы доступа: Public, private, protected, геттеры и сеттеры. Лекция-беседа. Освоить управление видимостью членов класса для реализации инкапсуляции, научиться проектировать безопасный внешний интерфейс класса, понимать роль аксессоров/мутаторов и современных синтаксических свойств. Модификаторы доступа: public, private, protected, internal/package-private. Инкапсуляция данных, сокрытие реализации. Геттеры, сеттеры, валидация при изменении состояния. Языковые свойства: C# properties, Python @property, Java-стиль, Kotlin val/var. Read-only и write-only поля, иммутабельные интерфейсы.</p> <p>Тема 2.4. Конструкторы и деструкторы: перегрузка конструкторов, управление жизненным циклом объекта. Лекция-беседа. Освоить механизмы создания и завершения жизни объектов, научиться проектировать гибкие инициализаторы с перегрузкой и цепочками вызовов, понимать принципы управления ресурсами и роль финализаторов/деструкторов. Конструктор: назначение, синтаксис, отличие от обычных методов. Перегрузка конструкторов, конструктор по умолчанию, параметризованный конструктор. Цепочки вызовов: this(...), super(...)/base(...).</p>

	Деструктор/финализатор: назначение, момент вызова, ограничения. Сборка мусора (GC) vs ручное управление памятью, RAII, using/with/defer. Статические конструкторы и блоки инициализации.
3.	<p>Раздел 3. Инкапсуляция</p> <p>Тема 3.1. Соккрытие данных: принципы инкапсуляции, разделение интерфейса и реализации. Лекция-беседа. Сформировать системное понимание инкапсуляции как механизма управления сложностью, научиться чётко разделять публичный контракт (что делает объект) и внутреннюю реализацию (как он это делает), осознать влияние сокращения данных на сопровождаемость, тестируемость и эволюцию кода. Соккрытие информации (Information Hiding, D.L. Parnas). Интерфейс vs реализация: контракт, инварианты, граничные условия. Связность (coupling) и зацепление: сильное, слабое, нулевое. Стабильность интерфейса при изменении внутренней структуры. «Утечка абстракции» (Leaky Abstraction) и её последствия.</p> <p>Тема 3.2. Лекция-беседа. Работа с приватными и защищенными членами класса: методы доступа, реализация геттеров и сеттеров. Освоить корректное применение модификаторов <code>private</code> и <code>protected</code> для управления видимостью, научиться реализовывать безопасные аксессоры/мутаторы с валидацией и бизнес-логикой, понимать ограничения геттеров/сеттеров и альтернативные подходы к взаимодействию с объектами. Уровни доступа: <code>private</code>, <code>protected</code>, пакетный/сборочный, <code>public</code>. Геттеры, сеттеры, валидаторы, инварианты класса. Защитное копирование (Defensive Copying) для ссылочных типов и коллекций. Принцип «Tell, Don't Ask»: поведенческие методы вместо извлечения состояния. Языковые свойства: C# <code>properties</code>, Python <code>@property</code>, Java-конвенции, Kotlin <code>val/var</code>.</p>
4.	<p>Раздел 4. Наследование</p> <p>Тема 4.1. Основы наследования: базовый и производный класс, расширение функциональности. Лекция-беседа. Сформировать представление о наследовании как механизме повторного использования кода и построения иерархий, освоить отношение «is-a» («является») как критерий обоснованности наследования, понимать цепочку инициализации и порядок создания объектов в иерархии. Базовый (родительский/супер) класс, производный (дочерний/под) класс. Отношение «is-a» vs «has-a», композиция и агрегация. Цепочка конструкторов, порядок инициализации полей. Наследование состояния и поведения, соккрытие членов (shadowing/hiding). Вводное знакомство с принципом подстановки Барбары Лисков (LSP).</p> <p>Тема 4.2. Типы наследования: одиночное наследование, множественное наследование. Лекция-беседа. Различать модели одиночного и множественного наследования, понимать их технические и архитектурные последствия, осознать природу «проблемы ромба» (Diamond Problem) и способы её разрешения, научиться заменять множественное наследование классов комбинацией одиночного наследования и интерфейсов/миксинов. Одиночное наследование (Java, C#, Kotlin), множественное наследование (C++, Python). Проблема ромба: амбигуитет вызовов, дублирование состояния, конфликты сигнатур. Порядок разрешения методов (MRO, C3-линеаризация в Python). Интерфейсы, traits, mixins как безопасные альтернативы. Композиция и делегирование как архитектурная замена наследованию.</p> <p>Тема 4.3. Переопределение методов: переопределение в производных классах, использование ключевого слова <code>super</code> или его аналогов. Лекция-беседа. Освоить механизм динамического связывания и переопределения (overriding) методов, научиться корректно использовать <code>super/base/parent</code> для вызова базовой реализации,</p>

	<p>различать переопределение, перегрузку (overloading) и скрытие (hiding/shadowing). Виртуальные методы, динамическая диспетчеризация, таблица виртуальных методов (vtable). Переопределение (override) vs перегрузка (overload) vs скрытие (new/shadow). Ключевые слова super/base/parent, порядок вызовов в цепочке. Аннотации контроля (@Override, override, final/sealed). Ковариантность возвращаемых типов, контравариантность параметров (вводно).</p> <p>Тема 4.4. Абстрактные классы и интерфейсы: частичная реализация классов, полиморфизм через интерфейсы. Лекция-беседа. Понимать различие между абстрактным классом (частичная реализация) и интерфейсом (чистый контракт), научиться проектировать расширяемые архитектуры на основе интерфейсов, применять полиморфизм через интерфейсы для слабой связности компонентов. Абстрактный класс, абстрактный метод, запрет инстанцирования. Интерфейс как контракт, множественная реализация интерфейсов. Default-методы в интерфейсах (Java 8+, C# 8+, PHP 8.0+). Принцип разделения интерфейсов (ISP) и инверсия зависимостей (DIP). Явные/неявные реализации интерфейсов, конфликты сигнатур.</p>
5.	<p>Раздел 5. Полиморфизм</p> <p>Тема 5.1. Статический и динамический полиморфизм: перегрузка методов, переопределение методов. Лекция-беседа. Различать статический (compile-time) и динамический (runtime) полиморфизм, освоить механизмы перегрузки и переопределения методов, понимать этапы разрешения вызовов, научиться выбирать тип полиморфизма в зависимости от задачи и требований к гибкости кода. Статический полиморфизм: перегрузка (overloading), раннее связывание, разрешение сигнатур. Динамический полиморфизм: переопределение (overriding), позднее связывание. Сигнатура метода, тип аргументов, правила продвижения типов (type promotion). Разрешение вызовов: этап компиляции vs этап выполнения. Ограничения: статические/приватные/финальные методы не переопределяются.</p> <p>Тема 5.2. Виртуальные функции: позднее связывание, использование абстрактных методов. Лекция-беседа. Понимать механизм виртуальных вызовов и таблицу виртуальных методов (vtable) на концептуальном уровне, освоить проектирование контрактов через абстрактные методы и классы, научиться управлять расширяемостью иерархий с помощью virtual, abstract, final/sealed. Виртуальный метод, таблица виртуальных методов (vtable), указатель vptr. Позднее (динамическое) связывание: подстановка реализации в runtime. Абстрактный метод, запрет инстанцирования, обязательная реализация. Модификаторы контроля: virtual/open, override, final/sealed, pure virtual. Производительность: накладные расходы виртуального вызова, devirtualization (оптимизация JIT/компилятора).</p> <p>Тема 5.3. Примеры применения полиморфизма: работа с коллекциями, объектов, реализация шаблонов проектирования. Лекция-беседа. Научиться использовать полиморфные коллекции для обработки разнородных объектов единым способом, распознавать и применять полиморфизм в типовых шаблонах проектирования, рефакторить условную логику (if/else, switch по типу) в полиморфный диспетчинг. Полиморфные коллекции: хранение объектов разных типов через общий базовый тип/интерфейс. Устранение instanceof/GetType() и каскадных проверок типов. Принцип открытости/закрытости (OCP) в практике полиморфизма. Паттерны, основанные на полиморфизме: Strategy, Command, State, Observer, Visitor. Dependency Injection и полиморфная подстановка зависимостей.</p>
6.	<p>Раздел 6. Абстракция</p> <p>Тема 6.1. Абстрактные классы: определение и использование, ограничения при работе с абстрактными классами. Лекция-беседа. Сформировать понимание абстрактного класса как механизма частичной реализации и принудительного</p>

	<p>наследования поведения, освоить правила объявления, инстанцирования и расширения абстрактных классов, научиться выявлять сценарии, где абстрактный класс предпочтительнее интерфейса или полностью конкретной реализации. Абстрактный класс, абстрактный метод, конкретный метод. Запрет прямого инстанцирования, роль в иерархии наследования. Цепочка конструкторов в абстрактных классах, статические члены. Ограничения: одиночное наследование классов, невозможность создания экземпляра, правила переопределения. Паттерн «Template Method» как классическое применение абстрактных классов.</p> <p>Тема 6.2. Интерфейсы: определение контрактов, реализация нескольких интерфейсов. Лекция-беседа. Понимать интерфейс как чистый контракт поведения без состояния, освоить синтаксис и семантику реализации нескольких интерфейсов в одном классе, научиться проектировать когезивные, слабо связанные интерфейсы с учётом принципа разделения (ISP). Интерфейс как контракт, сигнатуры методов, отсутствие полей состояния (в классическом ООП). Множественная реализация интерфейсов, разрешение конфликтов имён. Методы по умолчанию (default methods), статические методы в интерфейсах. Явная и неявная реализация (explicit/implicit implementation). Маркерные интерфейсы, функциональные интерфейсы, делегирование через контракты.</p> <p>Тема 6.3. Лекция-беседа. Примеры использования абстракции: проектирование сложных систем, упрощение взаимодействия между компонентами. Научиться выделять уровни абстракции для управления сложностью больших систем, освоить проектирование чётких границ взаимодействия (API, контракты, слоёная архитектура), развивать навык выявления и устранения «утечек абстракции» (leaky abstractions). Уровни абстракции: домен, сервис, инфраструктура, представление. Контракт-ориентированное проектирование (Contract-First). Принцип инверсии зависимостей (DIP), слабая связность. Фасадный паттерн (Facade), шлюзы (Gateways), адаптеры. Утечка абстракции: причины, симптомы, способы локализации. Абстракция в модульных и распределённых системах (микросервисы, плагины).</p>
7.	<p>Раздел 7. Обработка исключений</p> <p>Тема 7.1. Основы обработки ошибок: исключения как механизм управления ошибками, иерархия исключений. Лекция-беседа. Сформировать чёткое понимание различий между штатными возвратами, ошибками и исключениями, освоить чтение и анализ стека вызовов (stack trace), научиться классифицировать исключения и принимать осознанные решения: ловить или пробрасывать. Исключение (Exception) vs ошибка (Error) vs код возврата. Иерархия исключений: базовый класс, проверяемые (checked) и непроверяемые (unchecked/runtime). Стек вызовов (stack trace), точка выброса vs точка обработки. Проброс исключений (throw/rethrow), декларирование (throws/throw в сигнатуре). Принцип fail-fast и границы обработки ошибок.</p> <p>Тема 7.2. Блоки try-catch-finally: ловля и обработка исключений, освобождение ресурсов. Лекция-беседа. Освоить синтаксис и семантику блоков try-catch-finally и современных конструкций автоматического управления ресурсами, научиться гарантировать корректное освобождение ресурсов при любых сценариях выполнения, выявлять и избегать типовых антипаттернов обработки исключений. Порядок выполнения блоков try → catch → finally, возврат управления. Множественные catch, порядок от частного к общему, полиморфная обработка. Автоматическое управление ресурсами: try-with-resources (Java), using (C#), RAII (C++), контекстные менеджеры (Python). Подавленные исключения (suppressed</p>

	<p>exceptions), приоритет finally. Антипаттерны: проглатывание (catch {}), использование исключений для потока управления, пустые обработчики.</p> <p>Тема 7.3. Создание пользовательских исключений. Лекция-беседа. Научиться проектировать доменно-ориентированную иерархию исключений, освоить правила сохранения контекста и цепочки причин (exception chaining), понимать баланс между детализацией исключений и сложностью API. Наследование от базовых классов исключений, именование (суффикс Exception). Конструкторы: сообщение, код ошибки, причина (cause/innerException). Цепочка исключений (exception chaining), оборачивание низкоуровневых ошибок. Осмысленные сообщения, контекстные данные, локализуемость. Документирование выбрасываемых исключений (JSDoc, XML Doc, docstrings).</p>
8.	<p>Раздел 8. Работа с коллекциями</p> <p>Тема 8.1. Стандартные коллекции: списки, множества, словари, особенности работы с коллекциями в ООП. Лекция-беседа. Сформировать системное представление об иерархии интерфейсов коллекций и критериях их выбора, освоить работу с обобщениями (дженериками) для обеспечения типобезопасности, научиться управлять состоянием объектов внутри коллекций с учётом контрактов равенства и хеширования. Иерархия: List (индексированный доступ), Set (уникальность), Map/Dictionary (ключ-значение), Асимптотическая сложность операций: $O(1)$, $O(\log n)$, $O(n)$ для вставки, поиска, удаления. Контракт equals()/hashCode() (или __eq__/_hash__, IEquatable<T>). Иммутабельные и потокобезопасные коллекции, обёртки ReadOnly/Unmodifiable. Ковариантность, контравариантность и инвариантность обобщённых коллекций.</p> <p>Тема 8.2. Итераторы и генераторы: итерирование по коллекциям, создание собственных итераторов. Лекция-беседа. Освоить протокол итерации и механизмы обхода коллекций без привязки к внутренней реализации, научиться проектировать ленивые (lazy) источники данных с помощью генераторов, понимать trade-off между жадным (eager) и ленивым (lazy) вычислением. Интерфейсы Iterable/Iterator (hasNext(), next(), remove()). Синтаксис foreach/for-each и его трансляция в итераторы. Генераторы: yield return, yield*, контекстные менеджеры, состояние сопрограммы. Ленивая оценка vs жадная загрузка, буферизация, конвейерная обработка. Fail-fast vs fail-safe итераторы, модификация коллекции во время обхода.</p> <p>Тема 8.3. Шаблоны проектирования для работы с коллекциями. Лекция-беседа. Научиться заменять вложенные циклы и условную логику композицией паттернов и конвейеров обработки, освоить применение структурных и поведенческих паттернов к коллекциям и иерархиям, развивать навык проектирования fluent-интерфейсов и stream-подобных API. Паттерн Iterator: отделение обхода от структуры. Паттерн Composite: единая работа с деревом и листом. Паттерн Visitor: обход сложных иерархий без изменения классов. Strategy (компараторы/предикаты), Decorator (read-only, thread-safe, caching обёртки). Конвейерная обработка: Stream API / LINQ, map/filter/reduce, ленивые цепочки. Антипаттерны: «God Collection», raw-циклы для сложных преобразований, нарушение SRP в коллекциях.</p>
9.	<p>Раздел 9. Шаблоны проектирования</p> <p>Тема 9.1. Основные паттерны проектирования: порождающие паттерны (Singleton, Factory, Builder), структурные паттерны (Adapter, Decorator, Composite), поведенческие паттерны (Observer, Strategy, Command). Лекция-беседа. Сформировать системное представление о классификации GoF и назначении каждой группы паттернов, освоить структуру, намерение (intent), применимость и</p>

	<p>последствия использования базовых паттернов, научиться распознавать проблемы проектирования, которые естественно решаются конкретными паттернами. Классификация: порождающие, структурные, поведенческие. Порождающие: Singleton (потокобезопасность, антипаттерны), Factory Method / Abstract Factory (выбор реализации), Builder (пошаговое конструирование сложных объектов). Структурные: Adapter (совместимость интерфейсов), Decorator (динамическое добавление поведения), Composite (единая работа с деревом и листом). Поведенческие: Observer (реактивность, подписка/оповещение), Strategy (алгоритмы как interchangeable компоненты), Command (инкапсуляция запроса, undo/redo). UML-диаграммы классов/последовательностей для паттернов. Composition over inheritance как философия структурных решений.</p> <p>Тема 9.2. Примеры реализации паттернов: практическое применение в реальных проектах. Лекция-беседа. Научиться выявлять паттерны в коде популярных фреймворков и библиотек, освоить рефакторинг «спагетти-кода» в паттерн-ориентированную архитектуру, понимать принципы безопасной комбинации паттернов и оценки их стоимости/выгоды. Паттерны в индустрии: DI-контейнеры (Factory + Strategy), реактивные UI (Observer), pipeline-обработка (Decorator/Chain of Responsibility), event sourcing (Command). Комбинации паттернов: Factory + Strategy, Observer + Command, Decorator + Composite. Рефакторинг к паттернам: выделение интерфейсов, инкапсуляция изменчивости, устранение условного кода. Антипаттерны: Patternitis (использование ради использования), God Object, нарушение SRP через паттерны. Тестирование паттерн-ориентированного кода: моки, стабы, изоляция слоёв. Документирование архитектурных решений (ADR, C4 Model, диаграммы контекста).</p>
10.	<p>Раздел 10. Тестирование объектно-ориентированного кода</p> <p>Тема 10.1. Юнит-тестирование: написание тестов для классов и методов, использование фреймворков для тестирования. Лекция-беседа. Сформировать понимание роли юнит-тестов в жизненном цикле ООП-разработки. освоить структуру и дисциплину написания стабильных, воспроизводимых тестов, научиться использовать тестовые фреймворки и анализировать метрики качества кода. Юнит-тест, тестовый случай (test case), тестовый набор (test suite). Паттерн Arrange-Act-Assert (AAA): подготовка, действие, проверка. Фреймворки: JUnit, NUnit, pytest, xUnit, Google Test. Утверждения (assertions): эквивалентность, исключения, приближённые значения. Принципы FIRST: Fast, Independent, Repeatable, Self-validating, Timely. Покрытие кода (code coverage): строчное, ветвевое, условное, мутационное тестирование (вводно). Цикл TDD: Red → Green → Refactor.</p> <p>Тема 10.2. Моки и стабы: тестирование зависимостей, изоляция тестируемого кода. Лекция-беседа. Понимать различия между типами тестовых дублёров и сценариями их применения, освоить механизмы изоляции внешних зависимостей для фокусировки на поведении тестируемого класса, научиться проектировать классы с внедрением зависимостей (DI) для естественной тестируемости. Тестовые дублёры (Test Doubles): Dummy, Stub, Spy, Mock, Fake. Мокинг-фреймворки: Mockito, Moq, unittest.mock, NSubstitute. Внедрение зависимостей (Constructor/Property/Method DI), инверсия управления (IoC). Проверка состояния (state verification) vs проверка поведения (behavior verification). Антипаттерны: Mock Hell, избыточная изоляция, тестирование моков вместо реальной логики. Контрактное тестирование и границы модулей.</p>
11.	Раздел 11. Современные подходы к ООП

	<p>Тема 11.1. Фреймворки и библиотеки: примеры использования ООП в популярных фреймворках (например, Java Spring, .NET, Python Django). Лекция-беседа. Понимать, как классические принципы ООП воплощаются в архитектуре современных фреймворков, научиться анализировать фреймворковый код на предмет применения паттернов, DI и абстракций, освоить расширение фреймворков через наследование, композицию и интерфейсные контракты. Инверсия управления (IoC) и внедрение зависимостей (DI): контейнеры, жизненный цикл объектов. MVC/MVVM/Clean Architecture: разделение ответственности, слои абстракции. ORM и маппинг объектов: сущности как доменные модели, ленивая загрузка, прокси-объекты. AOP (Aspect-Oriented Programming): перехват вызовов, декораторы, аннотации/атрибуты. Convention over Configuration, рефлексия и code generation в фреймворках. Фреймворковые расширения: middleware, interceptors, plugins, custom handlers.</p> <p>Тема 11.2. ООП в многопоточном программировании: синхронизация потоков, безопасность данных. Лекция-беседа. Осознать риски работы с изменяемым состоянием объектов в конкурентной среде, освоить механизмы синхронизации и проектирования потокобезопасных классов, научиться выбирать современные подходы к параллелизму (async/await, акторы, иммутабельность). Гонки данных (race conditions), взаимоблокировки (deadlocks), livelocks, starvation. Прimitives синхронизации: synchronized/lock, ReentrantLock, Mutex, Semaphore, volatile/Atomic. Потокобезопасные коллекции: ConcurrentHashMap, BlockingQueue, CopyOnWriteArrayList. Акторная модель, message passing, изоляция состояния. async/await, thread pools, cancellation tokens, контексты выполнения. Иммутабельные объекты как основа thread-safe дизайна.</p> <p>Тема 11.3. ООП и функциональное программирование: комбинирование парадигм, примеры гибридных подходов. Лекция-беседа. Понимать комплементарность ООП и ФП, а не их противостояние. научиться применять гибридные паттерны: инкапсуляция состояния + чистые преобразования, освоить современные языковые конструкции, объединяющие обе парадигмы. Функциональное ядро + императивная оболочка (Functional Core, Imperative Shell). Неизменяемость (immutability), чистые функции, побочные эффекты (side effects). records/data classes, pattern matching, discriminated unions, Option/Result. Конвейеры обработки: Stream API, LINQ, itertools, ленивые вычисления. Higher-order functions, делегаты/лямбды как стратегия/шаблон метода. DDD + Event Sourcing + CQRS как гибридная ООП/ФП архитектура.</p>
--	--

4.3. Практические (семинарские) занятия

Темы практических занятий и их трудоемкость приведены в таблице 5.

Таблица 5 – Практические занятия и их трудоемкость

№ п/п	Темы практических занятий	Формы практических занятий	Трудоемкость, (час)	Из них практической подготовки, (час)	№ раздела дисциплины
Учебным планом не предусмотрено					
Всего					

4.4. Лабораторные занятия

Темы лабораторных занятий и их трудоемкость приведены в таблице 6.

Таблица 6 – Лабораторные занятия и их трудоемкость

№ п/п	Наименование лабораторных работ	Трудоемкость, (час)	Из них практической подготовки, (час)	№ раздела дисциплины
Семестр 6				
1	ЛР 1 Парадигмы программирования и основы классов	2	2	1, 2
2	ЛР 2 Инкапсуляция, конструкторы и RAII	2	2	2, 3
3	ЛР 3 Наследование и динамический полиморфизм	2	2	4, 5
4	ЛР 4 Абстракция, интерфейсы и полиморфные коллекции	2	2	4-6
5	ЛР 5 Исключения и стандартные коллекции STL	2	2	7, 8
6	ЛР 6 Шаблоны проектирования	2	2	9
7	ЛР 7 Тестирование и многопоточность	2	2	10-11
8	ЛР 8 Современные подходы: гибридная парадигма и C++	3	3	11
Всего		17	17	

4.5. Выполнение курсового проекта/ курсовой работы
Учебным планом не предусмотрено

4.6. Самостоятельная работа обучающихся

Виды самостоятельной работы и ее трудоемкость приведены в таблице 7.

Таблица 7 – Виды самостоятельной работы и ее трудоемкость

Вид самостоятельной работы	Всего, час	Семестр 6, час
1	2	3
Изучение теоретического материала дисциплины (ТО)	20	20
Курсовое проектирование (КП, КР)		
Расчетно-графические задания (РГЗ)		
Выполнение реферата (Р)		
Подготовка к текущему контролю успеваемости (ТКУ)	8	8
Домашнее задание (ДЗ)		
Контрольные работы заочников (КРЗ)		
Подготовка к промежуточной аттестации (ПА)	10	10
Всего:	38	38

5. Перечень учебно-методического обеспечения

для самостоятельной работы обучающихся по дисциплине (модулю)

Учебно-методические материалы для самостоятельной работы обучающихся указаны в п.п. разделов 6-11.

6. Перечень печатных и электронных учебных изданий

Перечень печатных и электронных учебных изданий приведен в таблице 8.

Таблица 8– Перечень печатных и электронных учебных изданий

Шифр/ URL адрес	Библиографическая ссылка	Количество экземпляров в библиотеке (кроме электронных экземпляров)
URL: https://znanium.com/catalog/product/557111	Васюткина, И. А. Технология разработки объектно-ориентированных программ на JAVA / Васюткина И.А. - Новосибирск :НГТУ, 2012. - 152 с.: ISBN 978-5-7782-1973-1.	
URL: https://znanium.com/catalog/product/2008803	Иванова, Г. С. Объектно-ориентированное программирование : учебник / Г. С. Иванова, Т. Н. Ничушкина ; под общ. ред. Г. С. Ивановой. - Москва : МГТУ им. Баумана, 2014. - 456 с. - ISBN 978-5-7038-3921-8.	
URL: https://znanium.ru/catalog/product/1926392	Хорев, П. Б. Объектно-ориентированное программирование с примерами на C# : учебное пособие / П.Б. Хорев. — Москва : ФОРУМ : ИНФРА-М, 2023. — 200 с. — (Высшее образование: Бакалавриат). - ISBN 978-5-00091-680-3.	
URL: https://znanium.com/catalog/product/1819676	Объектно-ориентированное программирование на C++ : учебник / И. В. Баранова, С. Н. Баранов, И. В. Баженова [и др.]. - Красноярск : Сиб. федер. ун-т, 2019. - 288 с. - ISBN 978-5-7638-4034-6.	
URL: https://znanium.com/catalog/product/1232338	Игнаткин, А. А. Объектно-ориентированное программирование : курс лекций / А. А. Игнаткин. - Москва : ИД МИСиС, 2005. - 149 с.	

7. Перечень электронных образовательных ресурсов информационно-телекоммуникационной сети «Интернет»

Перечень электронных образовательных ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины приведен в таблице 9.

Таблица 9 – Перечень электронных образовательных ресурсов информационно-телекоммуникационной сети «Интернет»

URL адрес	Наименование
https://pro.guap.ru/	Элементы электронного курса по дисциплине размещены внутри ЭИОС ГУАП «Интегрированная среда обучения»

8. Перечень информационных технологий

8.1. Перечень программного обеспечения, используемого при осуществлении образовательного процесса по дисциплине.

Перечень используемого программного обеспечения представлен в таблице 10.

Таблица 10– Перечень программного обеспечения

№ п/п	Наименование
1	Электронная информационно-образовательная среда ГУАП «Интегрированная среда обучения» (https://pro.guap.ru/) разработана сотрудниками ГУАП (введена в

	эксплуатацию приказом ГУАП от 06.06.2017 № 05-215/17), перечень модулей и их функциональное назначение изложены по ссылке https://guap.ru/it/system/iso
2	Официальный сайт образовательной организации в сети «Интернет» (https://guap.ru/), разработан сотрудниками ГУАП (введен в эксплуатацию Приказом ГУАП от 23.03.2023 № 05-145/23)
3	Microsoft Office 2019 (договор ГУАП, информация о лицензии представлена по ссылке https://guap.ru/it/system/iso/po)

8.2. Перечень информационно-справочных систем, используемых при осуществлении образовательного процесса по дисциплине

Перечень используемых информационно-справочных систем представлен в таблице 11.

Таблица 11– Перечень информационно-справочных систем

№ п/п	Наименование
1	Электронный каталог библиотеки ГУАП с доступом к базе полнотекстовых изданий (https://lib.guap.ru.), доступ через личный кабинет читателя библиотеки ГУАП
2	Научная электронная библиотека «eLIBRARY» (https://elibrary.ru/), доступ через личный кабинет читателя библиотеки ГУАП, а также по IP -адресам ГУАП
3	ЭБС «Лань» (https://e.lanbook.com/), доступ через личный кабинет читателя библиотеки ГУАП, а также по IP -адресам ГУАП
4	ЭБС Znanium (https://znanium.ru/), доступ через личный кабинет читателя библиотеки ГУАП, а также по IP -адресам ГУАП

9. Материально-техническая база

Состав материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине, представлен в таблице 12.

Таблица 12 – Состав материально-технической базы

№ п/п	Наименование составной части материально-технической базы	Номер аудитории (при необходимости)
1	Мультимедийная лекционная аудитория: Специализированная мебель; технические средства обучения, служащие для представления учебной информации большой аудитории; набор демонстрационного оборудования (Интерактивный мультисенсорный дисплей на перекатной стойке FocusTouch Диагональ 70" – 1 шт., ПЭВМ – 1 шт.); Обеспечен доступ в электронную информационно-образовательную среду ГУАП по локальной вычислительной сети или точке доступа WiFi.	21-21 (ул. Большая Морская, д.67, лит. А)
2	Лаборатория компьютерного моделирования: – специализированная мебель; – технические средства обучения, служащие для представления учебной информации; ПЭВМ - Дисплей интерактивный НТС- 1 шт. Лабораторное оборудование: ПЭВМ – «Место рабочее	31-04 (ул. Большая Морская, д.67, лит. А)

	автоматизированное» – 18 шт. Обеспечен доступ в электронную информационно-образовательную среду ГУАП по локальной вычислительной сети или точке доступа WiFi.	
--	--	--

10. Оценочные средства для проведения промежуточной аттестации

10.1. Состав оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине приведен в таблице 13.

Таблица 13 – Состав оценочных средств для проведения промежуточной аттестации

Вид промежуточной аттестации	Перечень оценочных средств
Дифференцированный зачет	Список вопросов; Тесты.

10.2. В качестве критериев оценки уровня сформированности (освоения) компетенций обучающимися применяется 5-балльная шкала оценки сформированности компетенций, которая приведена в таблице 14. В течение семестра может использоваться 100-балльная шкала модульно-рейтинговой системы Университета, правила использования которой, установлены соответствующим локальным нормативным актом ГУАП.

Таблица 14 – Критерии оценки уровня сформированности компетенций

Оценка компетенции 5-балльная шкала	Характеристика сформированных компетенций
«отлично» «зачтено»	Обучающийся: – глубоко и всесторонне усвоил программный материал; – уверенно, логично, последовательно и грамотно его излагает; – опираясь на знания основной и дополнительной литературы, тесно связывает усвоенные научные положения с практической деятельностью направления; – умело обосновывает и аргументирует выдвигаемые им идеи; – делает выводы и обобщения; – свободно владеет системой специализированных понятий. – правильно выполнил от 90% до 100% тестовых заданий**.
«хорошо» «зачтено»	Обучающийся: – твердо усвоил программный материал, грамотно и по существу излагает его, опираясь на знания основной литературы; – не допускает существенных неточностей; – увязывает усвоенные знания с практической деятельностью направления; – аргументирует научные положения; – делает выводы и обобщения; – владеет системой специализированных понятий. – правильно выполнил от 70% до 89% тестовых заданий**.
«удовлетворительно» «зачтено»	– обучающийся усвоил только основной программный материал, по существу излагает его, опираясь на знания только основной литературы; – допускает несущественные ошибки и неточности; – испытывает затруднения в практическом применении знаний направления; – слабо аргументирует научные положения; – затрудняется в формулировании выводов и обобщений; – частично владеет системой специализированных понятий. – правильно выполнил от 51% до 69% тестовых заданий**.

Оценка компетенции	Характеристика сформированных компетенций
5-балльная шкала	
«неудовлетворительно» «не зачтено»	<ul style="list-style-type: none"> – обучающийся не усвоил значительной части программного материала; – допускает существенные ошибки и неточности при рассмотрении проблем в конкретном направлении; – испытывает трудности в практическом применении знаний; – не может аргументировать научные положения; – не формулирует выводов и обобщений. – правильно выполнил менее 51% тестовых заданий**.

10.3. Типовые контрольные задания или иные материалы.

Вопросы (задачи) для экзамена представлены в таблице 15.

Таблица 15 – Вопросы (задачи) для экзамена

№ п/п	Перечень вопросов (задач) для экзамена	Код индикатора
	Учебным планом не предусмотрено	

Вопросы для дифф. зачета представлены в таблице 16.

Таблица 16 – Вопросы для дифф. зачета

№ п/п	Перечень вопросов для дифф. зачета	Код индикатора
1	Что такое объектно-ориентированное программирование (ООП)?	ПК-2.Д.4
2	Какие основные принципы ООП вы знаете?	ПК-2.Д.4
3	Что такое класс в ООП?	ПК-2.Д.4
4	Что такое объект?	ПК-2.Д.4
5	Что такое инкапсуляция?	ПК-2.Д.4
6	Что такое наследование?	ПК-2.Д.4
7	Что такое полиморфизм?	ПК-2.Д.4
8	Что такое абстракция?	ПК-2.Д.4
9	В чем разница между абстрактным классом и интерфейсом?	ПК-2.Д.4
10	Что такое модификаторы доступа? Перечислите их и объясните.	ПК-2.Д.4
11	Что такое конструктор?	ПК-2.Д.4
12	Что такое деструктор?	ПК-2.Д.4
13	Что такое статические методы и переменные?	ПК-2.Д.4
14	Что такое метод toString()?	ПК-2.Д.4
15	Что такое переопределение методов?	ПК-2.Д.4
16	Что такое перегрузка методов?	ПК-2.Д.4
17	Что такое композиция?	ПК-2.Д.4
18	Что такое агрегация?	ПК-2.Д.4
19	Что такое интерфейс?	ПК-2.Д.4
20	Что такое SOLID-принципы?	ПК-2.Д.4
21	Что такое абстрактный метод?	ПК-2.Д.4
22	Что такое коллекции?	ПК-2.Д.4
23	Что такое исключения?	ПК-2.Д.4
24	Что такое многопоточность?	ПК-2.Д.4
25	Что такое паттерны проектирования?	ПК-2.Д.4
26	Почему в ООП рекомендуется программировать на уровне интерфейсов коллекций, а не конкретных реализаций?	ПК-2.Д.4
27	Как обобщения предотвращают ошибки приведения типов на этапе компиляции?	ПК-2.Д.4

28	Когда ленивая обработка может оказаться хуже жадной?	ПК-2.Д.4
29	Какие накладные расходы несёт использование декораторов и ленивых цепочек?	ПК-2.Д.4
30	В каких случаях предпочтительнее использовать иммутабельную коллекцию вместо изменяемой?	ПК-2.Д.4

Перечень тем для выполнения курсового проекта/ курсовой работы представлены в таблице 17.

Таблица 17 – Перечень тем для выполнения курсового проекта / курсовой работы

№ п/п	Примерный перечень тем для выполнения курсового проекта/ курсовой работы
	Учебным планом не предусмотрено

Вопросы для проведения промежуточной аттестации в виде тестирования представлены в таблице 18.

Таблица 18 – Примерный перечень вопросов для тестов

№ п/п	Примерный перечень вопросов для тестов	Код индикатора
<p>1 тип. Задание комбинированного типа с выбором одного верного ответа из четырех предложенных и обоснованием выбора</p> <p>Инструкция: Прочитайте текст, выберите правильный ответ и запишите аргументы, обосновывающие выбор ответа</p>		
1	<p>Что такое класс в объектно-ориентированном программировании?</p> <p>А) Переменная, хранящая данные; Б) Шаблон для создания объектов; В) Функция, выполняющая определенные действия; Г) Структура данных, содержащая только методы.</p>	ПК-2.Д.4
2	<p>Что такое объект в ООП?</p> <p>А) Экземпляр класса; Б) Абстрактная структура данных; В) Метод класса; Г) Наследуемый интерфейс.</p>	ПК-2.Д.4
3	<p>Что означает наследование в ООП?</p> <p>А) Скрытие данных от внешнего мира; Б) Возможность использовать один и тот же метод в разных классах; В) Разделение программы на независимые модули; Г) Создание нового класса на основе существующего.</p>	ПК-2.Д.4
4	<p>Что такое абстракция в ООП?</p> <p>А) Скрытие деталей реализации; Б) Создание новых классов на основе существующих; В) Выделение ключевых характеристик объекта; Г) Использование одного интерфейса для разных объектов.</p>	ПК-2.Д.4
5	<p>Какой модификатор доступа делает члены класса доступными только внутри самого класса?</p> <p>А) protected;</p>	ПК-2.Д.4

	Б) private; В) public; Г) default.	
2 тип. Задание комбинированного типа с выбором нескольких вариантов ответа из предложенных и развернутым обоснованием выбора Инструкция: Прочитайте текст, выберите правильные варианты ответа и запишите аргументы, обосновывающие выбор ответов		
6	Какие преимущества дает инкапсуляция? А) Упрощение взаимодействия между объектами; Б) Снижение производительности программы; В) Защита данных от несанкционированного доступа; Г) Возможность изменения внутренней реализации без влияния на внешний интерфейс.	ПК-2.Д.4
7	Что такое полиморфизм в ООП? А) Создание новых классов на основе существующих; Б) Использование одного интерфейса для работы с различными типами данных; В) Ограничение доступа к данным внутри класса; Г) Возможность объекта принимать различные формы.	ПК-2.Д.4
8	Что такое абстрактный класс? А) Класс, который содержит хотя бы один абстрактный метод; Б) ; Класс, который используется только для хранения данных В) Класс, который содержит только статические методы; Г) Класс, который нельзя инстанцировать.	ПК-2.Д.4
9	Какие из следующих утверждений о интерфейсах верны? А) Интерфейс может содержать поля с начальными значениями; Б) Интерфейс определяет только сигнатуры методов; В) Класс может реализовать несколько интерфейсов; Г) Интерфейс может быть инстанцирован.	ПК-2.Д.4
10	Что такое композиция в ООП? А) Процесс объединения нескольких классов в один; Б) Принцип наследования; В) Отношение "часть-целое" между объектами; Г) Возможность использования одного объекта внутри другого.	ПК-2.Д.4
3 тип. Задание закрытого типа на установление соответствия Инструкция: Прочитайте текст и установите соответствие. К каждой позиции, данной в левом столбце, подберите соответствующую позицию в правом столбце		
11	Установите соответствие между терминами и их определениями: 1 Инкапсуляция А) Способность класса использовать свойства и методы другого класса. 2 Полиморфизм Б) Процесс создания нескольких экземпляров одного класса. 3 Наследование В) Механизм, позволяющий объекту иметь множество форм.	ПК-2.Д.4

	Г) Скрытие внутренней реализации объекта от внешнего мира.	
12	<p>Установите соответствие между терминами и их определениями:</p> <p>1 Класс А) Экземпляр класса, содержащий конкретные значения атрибутов.</p> <p>2 Объект Б) Шаблон для создания объектов, определяющий их свойства и поведение.</p> <p>3 Интерфейс В) Описание набора методов, которые должны быть реализованы в классе.</p> <p> Г) Структура данных, хранящая только примитивные типы.</p>	ПК-2.Д.4
13	<p>Установите соответствие между терминами и их определениями:</p> <p>1 Абстракция А) Выделение главных характеристик объекта, исключая второстепенные.</p> <p>2 Конструктор Б) Процесс преобразования одного типа данных в другой.</p> <p>3 Деструктор В) Метод, вызываемый при создании объекта для инициализации его состояния.</p> <p> Г) Метод, вызываемый при уничтожении объекта для освобождения ресурсов.</p>	ПК-2.Д.4
14	<p>Установите соответствие между терминами и их определениями:</p> <p>1 Метод А) Функция, определенная внутри класса для выполнения действий над объектом.</p> <p>2 Атрибут Б) Переменная, хранящая данные объекта.</p> <p>3 Поля В) Свойства класса, описывающие состояние объекта.</p> <p> Г) Тип данных, используемый для хранения ссылок на другие объекты.</p>	ПК-2.Д.4
15	<p>Установите соответствие между терминами и их определениями:</p> <p>1 Композиция А) Отношение "часть-целое", где часть может существовать независимо.</p> <p>2 Агрегация Б) Процесс объединения нескольких классов в один.</p> <p>3 Ассоциация В) Отношение "часть-целое", где часть не существует без целого.</p> <p> Г) Связь между объектами, описывающая их взаимодействие.</p>	ПК-2.Д.4
<p>4 тип. Задание закрытого типа на установление последовательности</p> <p>Инструкция: Прочитайте текст и установите последовательность. Запишите соответствующую последовательность букв слева направо</p>		
16	<p>Установите правильную последовательность этапов разработки класса:</p> <p>А) Определение методов класса;</p> <p>Б) Тестирование функциональности класса;</p> <p>В) Определение атрибутов класса;</p> <p>Г) Создание экземпляров класса.</p>	ПК-2.Д.4
17	<p>Установите порядок действий при создании объекта:</p> <p>А) Инициализация полей объекта;</p> <p>Б) Возврат ссылки на объект;</p> <p>В) Выделение памяти для объекта;</p>	ПК-2.Д.4

	Г) Вызов конструктора класса.	
18	Установите правильную последовательность этапов инкапсуляции: А) Определение приватных полей; Б) Защита данных от несанкционированного доступа; В) Создание публичных методов доступа (геттеров и сеттеров); Г) Использование класса в программе.	ПК-2.Д.4
19	Установите этапы работы с полиморфизмом: А) Вызов переопределенных методов через базовый класс; Б) Создание объектов производных классов; В) Создание производных классов с переопределенными методами; Г) Определение базового класса с виртуальными методами.	ПК-2.Д.4
20	Установите этапы работы с коллекциями объектов: А) Обработка объектов в коллекции; Б) Создание коллекции; В) Добавление объектов в коллекцию; Г) Удаление объектов из коллекции.	ПК-2.Д.4
5 тип. Задание открытого типа с развернутым ответом Инструкция: Прочитайте текст и запишите развернутый обоснованный ответ или напишите пропущенное слово/словосочетание		
21	Какие основные принципы ООП вы знаете?	ПК-2.Д.4
22	Что такое класс и объект в ООП?	ПК-2.Д.4
23	Какие виды полиморфизма существуют?	ПК-2.Д.4
24	Как работает конструктор в классе?	ПК-2.Д.4
25	В чем разница между композицией и агрегацией?	ПК-2.Д.4

Перечень тем контрольных работ по дисциплине обучающихся заочной формы обучения, представлены в таблице 19.

Таблица 19 – Перечень контрольных работ

№ п/п	Перечень контрольных работ
	Не предусмотрено

10.4. Методические материалы, определяющие процедуры оценивания индикаторов, характеризующих этапы формирования компетенций, содержатся в локальных нормативных актах ГУАП, регламентирующих порядок и процедуру проведения текущего контроля успеваемости и промежуточной аттестации обучающихся ГУАП.

11. Методические указания для обучающихся по освоению дисциплины

11.1. Методические указания для обучающихся по освоению лекционного материала.

Основное назначение лекционного материала – логически стройное, системное, глубокое и ясное изложение учебного материала. Назначение современной лекции в рамках дисциплины не в том, чтобы получить всю информацию по теме, а в освоении фундаментальных проблем дисциплины, методов научного познания, новейших достижений научной мысли. В учебном процессе лекция выполняет методологическую,

организационную и информационную функции. Лекция раскрывает понятийный аппарат конкретной области знания, её проблемы, дает цельное представление о дисциплине, показывает взаимосвязь с другими дисциплинами.

Планируемые результаты при освоении обучающимися лекционного материала:

- получение современных, целостных, взаимосвязанных знаний, уровень которых определяется целевой установкой к каждой конкретной теме;
- получение опыта творческой работы совместно с преподавателем;
- развитие профессионально-деловых качеств, любви к предмету и самостоятельного творческого мышления.
- появление необходимого интереса, необходимого для самостоятельной работы;
- получение знаний о современном уровне развития науки и техники и о прогнозе их развития на ближайшие годы;
- научиться методически обрабатывать материал (выделять главные мысли и положения, приходить к конкретным выводам, повторять их в различных формулировках);
- получение точного понимания всех необходимых терминов и понятий.

Лекционный материал может сопровождаться демонстрацией слайдов и использованием раздаточного материала при проведении коротких дискуссий об особенностях применения отдельных тематик по дисциплине.

Структура предоставления лекционного материала:

- лекционный материал представляется преподавателям устно;
- лекция состоит из трёх основных частей: вступительной, основной и заключительной;
- вступительная часть определяет название темы, план и цель лекции;
- в основной части лекции реализуется научное содержание темы, все главные узловые вопросы, проводится вся система доказательств с использованием наиболее целесообразных методических приёмов;
- заключительная часть имеет целью обобщать в кратких формулировках основные идеи лекции, логически завершая её как целостное творение;
- отдельные виды лекций могут иметь свои особенности как по содержанию, так и по структуре.

11.2. Методические указания для обучающихся по участию в семинарах.

Учебным планом не предусмотрено.

11.3. Методические указания для обучающихся по прохождению практических занятий

Учебным планом не предусмотрено.

11.4. Методические указания для обучающихся по выполнению лабораторных работ

В ходе выполнения лабораторных работ обучающийся должен углубить и закрепить знания, практические навыки, овладеть современной методикой и техникой эксперимента в соответствии с квалификационной характеристикой обучающегося. Выполнение лабораторных работ состоит из экспериментально-практической, расчетно-аналитической частей и контрольных мероприятий.

Выполнение лабораторных работ обучающимся является неотъемлемой частью изучения дисциплины, определяемой учебным планом, и относится к средствам, обеспечивающим решение следующих основных задач обучающегося:

- приобретение навыков исследования процессов, явлений и объектов, изучаемых в рамках данной дисциплины;

- закрепление, развитие и детализация теоретических знаний, полученных на лекциях;
- получение новой информации по изучаемой дисциплине;
- приобретение навыков самостоятельной работы с лабораторным оборудованием и приборами.

Задание и требования к проведению лабораторных работ

Список заданий представлен в п 4.4, таблица 6. Лабораторные работы следует выполнять в ходе прохождения курса, внимательно разбирая представленный методический материал преподавателем, с загрузкой выполненных работ в личный кабинет обучающегося в установленные в «Личном кабинете ГУАП» сроки для каждой работы.

Структура и форма отчета о лабораторной работе

Отчет о лабораторной работе должен включать в себя: титульный лист, формулировку задания, теоретические положения, используемые при выполнении лабораторной работы, описание процесса выполнения лабораторной работы, полученные результаты и выводы.

Требования к оформлению отчета о лабораторной работе

По каждой лабораторной работе выполняется отдельный отчет. Титульный лист оформляется в соответствии с шаблоном (образцом), приведенным на сайте ГУАП (www.guap.ru) в разделе «ГУАП/Нормативная документация/Документация/Для учебного процесса». Текстовые и графические материалы оформляются в соответствии с действующими ГОСТами и требованиями, приведенными на сайте ГУАП (www.guap.ru) в разделе «ГУАП/Нормативная документация/Документация/Для учебного процесса».

11.5. Методические указания для обучающихся по выполнению курсового проекта/курсовой работы

Учебным планом не предусмотрено.

11.6. Методические указания для обучающихся по прохождению самостоятельной работы

В ходе выполнения самостоятельной работы, обучающийся выполняет работу по заданию и при методическом руководстве преподавателя, но без его непосредственного участия.

В процессе выполнения самостоятельной работы у обучающегося формируется целесообразное планирование рабочего времени, которое позволяет ему развивать умения и навыки в усвоении и систематизации приобретаемых знаний, обеспечивает высокий уровень успеваемости в период обучения, помогает получить навыки повышения профессионального уровня.

Методическими материалами, направляющими самостоятельную работу обучающихся, являются:

- учебно-методический материал по дисциплине.

11.7. Методические указания для обучающихся по прохождению текущего контроля успеваемости.

Текущий контроль успеваемости предусматривает контроль качества знаний обучающихся, осуществляемого в течение семестра с целью оценивания хода освоения дисциплины.

Проведение текущего контроля успеваемости осуществляется с помощью тестов, приведенных в таблице 18. Оценивание текущего контроля успеваемости оценивается по

четырёх бальной системе: «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

11.8. Методические указания для обучающихся по прохождению промежуточной аттестации.

Промежуточная аттестация обучающихся предусматривает оценивание промежуточных и окончательных результатов обучения по дисциплине. Она включает в себя:

- дифференцированный зачет – это форма оценки знаний, полученных обучающимся при изучении дисциплины, при выполнении курсовых проектов, курсовых работ, научно-исследовательских работ и прохождении практик с аттестационной оценкой «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Промежуточная аттестация оценивается по результатам текущего контроля успеваемости. В случае, если студент по уважительной причине не выполнил требования текущего контроля, ему предоставляется возможность сдать задолженности по пропущенным темам. Форма проведения промежуточной аттестации – письменная.

Дифференцированный зачет обучающийся получает при выполнении и сдаче не менее 80% лабораторных работ, выполненных в полном объеме, пройденному и сданному тестированию текущего контроля с оценкой не ниже «удовлетворительно».

Лист внесения изменений в рабочую программу дисциплины

Дата внесения изменений и дополнений. Подпись внесшего изменения	Содержание изменений и дополнений	Дата и № протокола заседания кафедры	Подпись зав. кафедрой