МИНИСТЕРСТВО ОБРАЗОВАНИЯ СВЕРДЛОВСКОЙ ОБЛАСТИ

ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ СВЕРДЛОВСКОЙ ОБЛАСТИ
«ЕКАТЕРИНБУРГСКИЙ МОНТАЖНЫЙ КОЛЛЕДЖ»

Рабочая программа учебной дисциплины
«ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ»
для специальности
09.02.07 «Информационные системы и программирование»

Екатеринбург

2025

Чистяков Валерий Николаевич
я подтверждаю этот документ своей удостоверяющей подписью
место подписания
2026.04.08 17:02:08+05'00'

Рабочая программа учебной дисциплины **«Объектно-ориентированное программирование»** разработана на основе Федерального государственного образовательного стандарта среднего профессионального образования по специальности **09.02.07 «Информационные системы и программирование».**

Организация-разработчик: ГАПОУ СО «Екатеринбургский монтажный колледж»

Разработчик: Черноскутов Д.А., преподаватель ГАПОУ СО «Екатеринбургский монтажный колледж».

Рассмотрена методическим объединением Информационных технологий

Руководитель МО _____Софьина Н.А.
7.05.25

УТВЕРЖДАЮ
Заместитель директора по учебной работе
_____ Л.С.Хоринова
«13» мая 2025 г.

СОДЕРЖАНИЕ

1. **ОБЩАЯ ХАРАКТЕРИСТИКА РАБОЧЕЙ ПРОГРАММЫ УЧЕБНОЙ ДИСЦИПЛИНЫ**
2. **СТРУКТУРА И СОДЕРЖАНИЕ РАБОЧЕЙ ПРОГРАММЫ УЧЕБНОЙ ДИСЦИПЛИНЫ**
3. **УСЛОВИЯ РЕАЛИЗАЦИИ РАБОЧЕЙ ПРОГРАММЫ УЧЕБНОЙ ДИСЦИПЛИНЫ**
4. **КОНТРОЛЬ И ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ УЧЕБНОЙ ДИСЦИПЛИНЫ**

1. ОБЩАЯ ХАРАКТЕРИСТИКА РАБОЧЕЙ ПРОГРАММЫ УЧЕБНОЙ ДИСЦИПЛИНЫ «ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ»

1.1. Область применения рабочей программы

Рабочая программа учебной дисциплины «Объектно-ориентированное программирование» является частью основной образовательной программы в соответствии с ФГОС СПО 09.02.07 «Информационные системы и программирование».

1.2. Место дисциплины в структуре основной образовательной программы:

Общепрофессиональная дисциплина профессионального цикла (ОП.15)

1.3. Цель и планируемые результаты освоения дисциплины:

В рамках программы учебной дисциплины обучающимися осваиваются умения и знания

Код ПК, ОК	Умения	Знания
ОК 1 Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам.	– разрабатывать алгоритмы для конкретных задач; – составлять простые блок-схемы алгоритмов; – использовать программы для графического отображения алгоритмов; – определять сложность работы алгоритмов. – работать в интегрированной среде изучаемых языков программирования; – реализовывать построенные алгоритмы в виде программ на конкретном языке программирования.	– свойства алгоритмов, общие принципы построения алгоритмов; – основные алгоритмические конструкции; – основные элементы языка, структуры программы, операторы и операции, управляющие структур, структуры данных;

ОК 2 Использовать современные средства поиска, анализа и интерпретации информации, и информационные технологии для выполнения задач профессиональной деятельности.	– определять задачи поиска и источники информации; – планировать процесс поиска; – выявлять и эффективно искать информацию, необходимую для решения задачи и/или проблемы; – структурировать получаемую информацию, выделять значимое для профессиональной деятельности; – оформлять результаты поиска – анализировать задачу и/или проблему и выделять её составные части; – определять этапы решения задачи; –применять современные средства информационных технологий для решения профессиональных задач;	– основные приемы структурирования информации; – основные источники информации и ресурсы для решения задач и проблем в профессиональном и/или социальном контексте; – формат оформления результатов поиска
ОК 3 Планировать и реализовывать собственное профессиональное и личностное развитие, предпринимательскую деятельность в профессиональной сфере, использовать знания по финансовой грамотности в различных жизненных ситуациях.	– определять и выстраивать траектории профессионального и личностного развития	– стандарты выполнения работ и методов работы для решения профессиональных задач – структуры плана для решения задач;
ОК 4 Эффективно взаимодействовать и работать в коллективе и команде, руководством, клиентами.	– владеть этикой делового общения; – эффективно общаться с сокурсниками, преподавателями, представителями работодателя;	– основы организации работы коллектива исполнителей; – принципы делового общения в коллективе
ОК 5. Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста.	– применять современную научную профессиональную терминологию;	– современную научную и профессиональную терминологию;

ПК 5.2. Разрабатывать проектную документацию на разработку информационной системы в соответствии с требованиями заказчика.	– формировать алгоритмы разработки программных модулей в соответствии с техническим заданием; – составить план действия; – определить необходимые ресурсы;	– стандарты кодирования; – основные этапы разработки программного обеспечения; – основные принципы разработки программных продуктов, основанные на объектно-ориентированном программировании;
ПК 5.3. Разрабатывать подсистемы безопасности информационной системы в соответствии с техническим заданием.	– использовать языки структурного, объектно-ориентированного программирования для создания программ; – осуществлять разработку кода программы на языках низкого и высокого уровней; – использовать существующий код в качестве основы для анализа и модификации;	–основные этапы разработки программного обеспечения; – основные принципы объектно-ориентированного программирования
ПК 5.4. Производить разработку модулей информационной системы в соответствии с техническим заданием	– работать в интегрированной среде изучаемых языков программирования; – разрабатывать алгоритмы для конкретных задач; – создавать программу по разработанному алгоритму как отдельный модуль; – осуществлять разработку кода программного модуля на современных языках программирования; – применять основные приемы объектно-ориентированного решения задач и – способы построения объектно-ориентированных алгоритмов;	– основные алгоритмические конструкции; – основные положения концепции объектно-ориентированного программирования; – теоретические основы, методы, способы и приемы объектно-ориентированного программирования; – способы проектирования объектно-ориентированных программных систем; – архитектуру и возможности современных языков объектно-ориентированного программирования; – интегрированную среду разработки для изучаемого языка программирования;

ПК 5.5. Осуществлять тестирование информационной системы на этапе опытной эксплуатации с фиксацией выявленных ошибок кодирования в разрабатываемых модулях информационной системы.	– отлаживать и проводить испытания объектно-ориентированных программ;	– основные принципы отладки и тестирования программных продуктов

2. СТРУКТУРА И СОДЕРЖАНИЕ РАБОЧЕЙ ПРОГРАММЫ УЧЕБНОЙ ДИСЦИПЛИНЫ

2.1. Объем учебной дисциплины и виды учебной работы

Вид учебной работы	**Объем часов**
Максимальная учебная нагрузка	**60**
Обязательная учебная нагрузка	**50**
в том числе:	
теоретическое обучение	20
практические занятия	30
Самостоятельная работа	**10**
Итоговая аттестация проводится в форме дифференцированного зачета	

2.2. Тематический план и содержание учебной дисциплины

Наименование разделов и тем	Содержание учебного материала, лабораторные работы и практические занятия, самостоятельная учебная работа обучающихся, курсовая работа (проект) *(если предусмотрены)*	Объем часов	Коды компетенций, формированию которых способствует элемент программы
1	2	3	4
Тема 1. Основы объектно-ориентированного подхода.	**Содержание**	**6**	ОК 1 - ОК 5, ПК 5.2 - ПК 5.5
	1. Основные направления в программировании. Возникновение объектно-ориентированного программирования (ООП). Базовые принципы ООП	2	
	2. Парадигма программирование. Фундаментальные методы и свойства объектной модели, ее преимущества.	2	
	3. Основные положения объектной модели, ее преимущества. Абстрагирование. Полиморфизм. Наследственность. Модульное программирование. Иерархия. Типизация.	2	
Тема 2. Базовые абстракции объектно-ориентированного анализа и программирования	**Содержание**	14	ОК 1 - ОК 5, ПК 5.2 - ПК 5.5
	1. Базовые конструкции объектно-ориентированных программ: классы и объекты. Инициализация и разрушение объекта. Компоненты класса. Конструкторы и деструкторы. Перегрузка и переопределение методов класса.	2	
	2. Методы и механизмы инкапсуляции и организации доступа к элементам объекта. Принцип инкапсуляции.	2	
	3. Область действия класса и доступ к компонентам класса. Управление доступом к компонентам класса.	2	
	4. Структурные элементы класса и методы взаимодействия объектов. Организация внешнего доступа к локальным компонентам класса. Интерфейсные (дружественные) методы. Статические и константные компоненты.	2	
	5. Параметризация объектов в ООП. Введение в параметризированные классы. Параметризированные классы и методы, их свойства. Совместное использование параметризации и принципов наследования.	2	
	6. Организация внешнего доступа к компонентам параметризированных классов. Параметризированные классы и статические члены.	2	
	7. Статические и динамические объекты. Указатели и ссылки. Операторы для динамического выделения и освобождения памяти. Статические и динамические объекты. Proxy-классы.	2	

	Тематика практических занятий и лабораторных работ	**30**	
	1. Основы языка Python. Подмножество.	**2**	
	2. Классы и объекты. Создание диаграммы классов	**2**	
	3. Классы и объекты.	**2**	
	4. Методы и механизмы инкапсуляции и организации доступа к элементам объекта.	**2**	
	5. Методы и механизмы инкапсуляции и организации доступа к элементам объекта.	**2**	
	6. Структурные элементы класса, методы взаимодействия объектов.	**2**	
	7. Структурные элементы класса, методы взаимодействия объектов.	**2**	
	8. Статические и динамические объекты.	**2**	
	9. Статические и динамические объекты.	**2**	
	10.Методы и механизмы наследования и определения собственных типов данных.	**2**	
	11.Методы и механизмы наследования и определения собственных типов данных.	**2**	
	12.Полиморфизм, использование виртуального механизма для реализации принципа полиморфизма.	**2**	
	13.Полиморфизм, использование виртуального механизма для реализации принципа полиморфизма.	**2**	
	14.Параметризация объектов в ООП.	**2**	
	15.Использование параметризированных классов.	**2**	
	Самостоятельная работа обучающихся	**10**	
	Создать презентацию «Классы, их структуры и возможности». Создать презентацию «Разработка простейших классов» Создать презентацию «Класс Object». Составить таблицу «Иерархия класса»	**10**	
Всего:		**60**	

3. УСЛОВИЯ РЕАЛИЗАЦИИ РАБОЧЕЙ ПРОГРАММЫ УЧЕБНОЙ ДИСЦИПЛИНЫ

3.1. Для реализации программы профессионального модуля должны быть предусмотрены следующие специальные помещения:

Реализация программы предполагает наличие стандартного учебного кабинета и лаборатории программирования.

Оборудование лаборатории и рабочих мест лаборатории:

- АРМ обучающихся;
- АРМ преподавателя;
- маркерная доска;
- комплект учебно-методической документации.
- персональные компьютеры с лицензионным программным обеспечением, операционной системой и выходом в Интернет;
- мультимедиапроектор и экран.

3.2. Информационное обеспечение реализации программы

Для реализации программы библиотечный фонд ГАПОУ СО «ЕМК» обладает следующим перечнем используемых учебных изданий, Интернет-ресурсов, дополнительной литературы

3.2.1. Основная литература

1. Голицына, О. Л. Основы алгоритмизации и программирования: учебное пособие / О.Л. Голицына, И.И. Попов. - 4-е изд., испр. и доп. - Москва: ФОРУМ: ИНФРА-М, 2021. - 431 с. - (Среднее профессиональное образование). - ISBN 978-5-00091-570-7. - Текст: электронный. - URL: https://znanium.com/catalog/product/1150328 – Режим доступа: по подписке.

2. Гуриков, С.Р. Введение в программирование на языке Visual C#: учебное пособие / С.Р. Гуриков. - Москва: ФОРУМ: ИНФРА-М, 2019. - 447 с. - (Среднее профессиональное образование). - ISBN 978-5-00091-540-0. - Текст: электронный. - URL: https://znanium.com/catalog/product/1012397. – Режим доступа: по подписке.

3. Дорогов, В. Г. Основы программирования на языке С: учебное пособие / В.Г. Дорогов, Е.Г. Дорогова; под ред. Л.Г. Гагариной. - Москва: ФОРУМ: ИНФРА-М, 2020. - 224 с. - (Среднее профессиональное образование). - ISBN 978-5-8199-0809-9. - Текст: электронный. - URL: https://znanium.com/catalog/product/1082440. – Режим доступа: по подписке.

4. Канцедал, С.А. Алгоритмизация и программирование: учебное пособие / С.А. Канцедал. - Москва: ФОРУМ: ИНФРА-М, 2021. - 352 с. - (Среднее профессиональное образование). - ISBN 978-5-8199-0727-6. - Текст: электронный. - URL: https://znanium.com/ catalog/product/1189320. – Режим доступа: по подписке.

5. Колдаев, В.Д. Основы алгоритмизации и программирования: учебное пособие / В.Д. Колдаев; под ред. проф. Л.Г. Гагариной. - Москва: ФОРУМ: ИНФРА-М, 2022. - 414 с. - (Среднее профессиональное образование). - ISBN 978-5-8199-0733-7. - Текст: электронный. - URL: https://znanium.com/catalog/product/1735805. - Режим доступа: по подписке.

6. Кузин, А.В. Программирование на языке Си: учебное пособие / А.В. Кузин, Е.В. Чумакова. - Москва: ФОРУМ: ИНФРА-М, 2020. - 143 с. - (Среднее профессиональное образование). - ISBN 978-5-00091-556-1. - Текст: электронный. - URL: https://znanium.com/catalog/product/961653. – Режим доступа: по подписке.

7. Хорев, П.Б. Объектно-ориентированное программирование с примерами на C#: учебное пособие / П.Б. Хорев. - Москва: ФОРУМ: ИНФРА-М, 2021. - 200 с. - (Среднее профессиональное образование). - ISBN 978-5-00091-713-8. - Текст: электронный. - URL: https://znanium.com/catalog/product/1195623. – Режим доступа: по подписке.

3.2.2. Основные электронные издания

1. METANIT.COM. Сайт о программировании. Режим доступа: https://metanit.com/

2. Microsoft .NET Руководство по языку C#. Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp/

3. Электронный курс на системе дистанционного обучения ГАПОУ СО «ЕМК»: http://188.234.244.32/course/view.php?id=569

3.2.3. Дополнительные источники

1. TutorialsTeacher. Режим доступа - https://www.tutorialsteacher.com/csharp/csharp-tutorials

2. Основы программирования на примере C#. Часть 1. Режим доступа - https://ulearn.me/Course/BasicProgramming/Kratkaya_spravka_pered_nachalom_69a2e121-e58f-4cd0-8221-7affb7dc796e

3. Основы программирования на примере C#. Часть 2. Режим доступа - https://ulearn.me/Course/BasicProgramming2/Steki_i_ocheredi_48016626-87ae-411d-ae97-f7a49e465dbc

4. Программирование на C#. Режим доступа - https://openedu.ru/course/urfu/CSHARP/?session=fall_2022

5. Проектирование на языке C#. Режим доступа - https://ulearn.me/course/cs2/C_i_Python_PHP_60871770-3d77-4d4c-b68f-ba9e3494d134

6. Технологии программирования. Режим доступа - https://openedu.ru/course/urfu/PRGRMM/?session=fall_2022

4. КОНТРОЛЬ И ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ УЧЕБНОЙ ДИСЦИПЛИНЫ

Результаты обучения	Критерии оценки	Формы и методы оценки
Перечень знаний, осваиваемых в рамках дисциплины: – современная научная и профессиональная терминология; – основные приемы структурирования информации; – структура составления плана для решения задач; – основные источники информации и ресурсы для решения задач и проблем в профессиональном и/или социальном контексте; – формат оформления результатов поиска; – стандарты выполнения работ и методы работы для решения профессиональных задач; – основные алгоритмические конструкции; – основные элементы языка, структура программы, операторы и операции, управляющие структуры, структуры данных; – стандарты кодирования; – основные этапы разработки программного обеспечения; – основные принципы разработки программных продуктов, основанные на объектно-ориентированном программировании; – основные этапы разработки программного обеспечения; – основные принципы объектно-ориентированного программирования – основные положения концепции объектно-ориентированного программирования; – теоретические основы, методы, способы и приемы объектно-ориентированного программирования; – способы проектирования объектно-ориентированных программных систем; – архитектуру и возможности современных языков объектно-ориентированного программирования; – интегрированная среда разработки для изучаемого языка программирования;	Практические и самостоятельные работы, дифференцированный зачет: «5» - 90-100% правильно выполненного задания; «4» - 80-89% правильно выполненного задания; «3» - выполнение практически всей работы (не менее 70%) «2» - выполнение менее 70% всей работы.	Тесты Практические занятия Задания внеаудиторной самостоятельной работы Дифференциро-ванный зачет

– основные принципы отладки и тестирования программных продуктов; – основы организации работы коллектива исполнителей; – принципы делового общения в коллективе		
Перечень умений, осваиваемых в рамках дисциплины: – применять современную научную профессиональную терминологию; – анализировать задачу и/или проблему и выделять её составные части; – определять этапы решения задачи; – определять задачи поиска и источники информации; – планировать процесс поиска; – выявлять и эффективно искать информацию, необходимую для решения задачи и/или проблемы; – структурировать получаемую информацию, выделять значимое для профессиональной деятельности; – оформлять результаты поиска; – разрабатывать алгоритмы для конкретных задач; – применять современные средства информационных технологий для решения профессиональных задач; – работать в интегрированной среде изучаемых языков программирования; – формировать алгоритмы разработки программных модулей в соответствии с техническим заданием; – создавать программу по разработанному алгоритму как отдельный модуль; – использовать языки структурного, объектно-ориентированного программирования для создания программ; – применять основные приемы объектно-ориентированного решения задач и способы построения объектно-ориентированных алгоритмов; – осуществлять разработку кода программы на языках низкого и высокого уровней; – отлаживать и проводить испытания объектно-ориентированных программ; – использовать существующий код в качестве основы для анализа и модификации;		

– определять и выстраивать траектории профессионального и личностного развития; – владеть этикой делового общения; – эффективно общаться с сокурсниками, преподавателями, представителями работодателя.		

МИНИСТЕРСТВО ОБРАЗОВАНИЯ СВЕРДЛОВСКОЙ ОБЛАСТИ

ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ СВЕРДЛОВСКОЙ ОБЛАСТИ «ЕКАТЕРИНБУРГСКИЙ МОНТАЖНЫЙ КОЛЛЕДЖ»

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ
для специальности
09.02.07 «Информационные системы и программирование»

Екатеринбург
2025

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Фонд оценочных средств (ФОС) предназначен для организации и контроля учебного процесса по учебной дисциплине «ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ», входящему в профессиональный цикл подготовки студентов по направлению разработки.

ФОС включает перечень **лекционных и практических тем**, а также **краткое содержание занятий**, раскрывающее основное содержание и направленность работы обучающихся на каждом этапе.

Фонд разработан с целью:

- обеспечения **прозрачности и системности оценки результатов обучения**;
- помощи преподавателям при планировании и проведении занятий;
- информирования студентов о тематике, задачах и ожидаемых результатах по каждой теме курса;
- формирования **базы заданий для текущего контроля знаний** (практические работы, анализ, аудиты.);
- подготовки к **промежуточной аттестации в форме зачета**.

Материалы ФОС могут использоваться:

- при проведении **текущего, рубежного и итогового контроля**;
- как основа для составления **оценочных листов, чек-листов** и других форм диагностики;
- при организации **самостоятельной и проектной деятельности** студентов.

Завершается изучение дисциплины **зачетом.**

Фонд является важным инструментом обеспечения качества подготовки специалистов и соответствует требованиям ФГОС СПО.

Описание лекционных занятий

1. **Основные направления в программировании. Возникновение объектно-ориентированного программирования (ООП). Базовые принципы ООП.**

Рассматриваются основные этапы развития программирования, переход от процедурного к объектно-ориентированному подходу. Изучаются предпосылки возникновения ООП, его принципы и преимущества: инкапсуляция, наследование и полиморфизм.

2. **Парадигма программирование. Фундаментальные методы и свойства объектной модели, ее преимущества.**

Раскрывается понятие парадигмы программирования. Анализируются фундаментальные свойства объектной модели: модульность, абстракция, иерархичность и повторное использование кода. Определяются преимущества объектного подхода.

3. **Основные положения объектной модели, ее преимущества. Абстрагирование. Полиморфизм. Наследственность. Модульное программирование. Иерархия. Типизация.**

Изучаются ключевые положения объектной модели и их значение в построении программных систем. Рассматриваются абстрагирование, полиморфизм, наследование, модульное программирование, иерархия классов и типизация.

4. **Базовые конструкции объектно-ориентированных программ: классы и объекты. Инициализация и разрушение объекта. Компоненты класса. Конструкторы и деструкторы. Перегрузка и переопределение методов класса.**

Рассматриваются структура класса и объекта, их жизненный цикл. Изучаются конструкторы и деструкторы, а также механизмы перегрузки и переопределения методов для реализации гибкости поведения объектов.

5. **Методы и механизмы инкапсуляции и организации доступа к элементам объекта. Принцип инкапсуляции.**

Рассматривается принцип инкапсуляции как основа объектной модели. Изучаются механизмы управления доступом к данным, модификаторы доступа и способы защиты компонентов класса.

6. **Область действия класса и доступ к компонентам класса. Управление доступом к компонентам класса.**

Изучаются области видимости и способы доступа к членам класса. Рассматриваются уровни доступа, принципы ограничения видимости и управление взаимодействием между объектами.

7. **Структурные элементы класса и методы взаимодействия объектов. Организация внешнего доступа к локальным компонентам класса. Интерфейсные (дружественные) методы. Статические и константные компоненты.**

Анализируются структурные элементы класса и методы их взаимодействия. Изучается применение дружественных методов, статических и константных членов для организации эффективной структуры программы.

8. **Параметризация объектов в ООП. Введение в параметризированные классы. Параметризированные классы и методы, их свойства. Совместное использование параметризации и принципов наследования.**

Вводится понятие параметризации. Рассматриваются шаблонные классы и методы, их структура, свойства и сочетание с наследованием для реализации универсальных решений.

9. **Организация внешнего доступа к компонентам параметризированных классов. Параметризированные классы и статические члены.**

Рассматриваются особенности организации внешнего доступа к параметризированным классам. Объясняется использование статических членов в обобщённых структурах и их влияние на общие свойства класса.

10. **Статические и динамические объекты. Указатели и ссылки. Операторы для динамического выделения и освобождения памяти. Статические и динамические объекты. Proxy-классы.**

Изучаются различия между статическими и динамическими объектами, работа с указателями и ссылками. Рассматриваются механизмы управления памятью и применение proxy-классов для контроля доступа к объектам.

Описание лабораторных занятий

1. **Основы языка Python. Подмножество.**

Осваиваются базовые конструкции языка Python: типы данных, операторы, управляющие структуры и функции. Формируются навыки работы с подмножеством языка, необходимым для дальнейшего изучения ООП.

2. **Классы и объекты. Создание диаграммы классов.**

Практикуется создание собственных классов и объектов. Выполняется построение диаграммы классов для визуализации структуры программы и взаимосвязей между объектами.

3. **Классы и объекты.**

Продолжается работа с классами и объектами. Отрабатываются навыки определения атрибутов и методов, взаимодействия между объектами, а также практические примеры применения принципов ООП.

4. **Методы и механизмы инкапсуляции и организации доступа к элементам объекта.**

Изучаются механизмы инкапсуляции в Python. Практикуется использование модификаторов доступа и организация безопасного взаимодействия с внутренними компонентами класса.

5. **Методы и механизмы инкапсуляции и организации доступа к элементам объекта.**

Продолжается закрепление навыков применения инкапсуляции. Выполняются упражнения по созданию защищённых свойств и методов, реализации интерфейсов доступа к данным.

6. **Структурные элементы класса, методы взаимодействия объектов.**

Отрабатывается создание структурных элементов классов, реализация взаимодействия объектов через методы. Рассматриваются примеры проектирования классов с чётким распределением ролей и обязанностей.

7. **Структурные элементы класса, методы взаимодействия объектов.**

Закрепляются навыки организации взаимодействия между объектами. Практикуется использование дружественных методов и разработка простых интерфейсов для обмена данными между классами.

8. **Статические и динамические объекты.**

Рассматривается создание и использование статических и динамических объектов. Изучаются принципы хранения данных и различия между статическими переменными и экземплярами классов.

9. **Статические и динамические объекты.**

Закрепляются навыки работы со статическими и динамическими объектами. Практикуются операции выделения и освобождения памяти, использование ссылок и управление временем жизни объектов.

10. **Методы и механизмы наследования и определения собственных типов данных.**

Изучаются основы наследования в Python. Выполняются упражнения по созданию производных классов, переопределению методов и формированию собственных типов данных.

11. **Методы и механизмы наследования и определения собственных типов данных.**

Продолжается работа с наследованием. Практикуется множественное наследование, использование базовых и производных классов, расширение функциональности существующих структур.

12. **Полиморфизм, использование виртуального механизма для реализации принципа полиморфизма.**

Изучаются основы полиморфизма. Выполняются упражнения по созданию методов с одинаковыми именами, но различным поведением в разных классах, реализуется принцип подстановки.

13. **Полиморфизм, использование виртуального механизма для реализации принципа полиморфизма.**

Закрепляются навыки реализации полиморфизма. Практикуется использование абстрактных классов и интерфейсов, разработка программных решений с элементами динамической диспетчеризации методов.

14. **Параметризация объектов в ООП.**

Изучается механизм параметризации. Практикуется создание обобщённых классов и функций, использование параметров типов для повышения гибкости кода.

15. **Использование параметризированных классов.**

Закрепляются навыки применения параметризированных классов. Выполняются практические задания по реализации универсальных решений и комбинированию шаблонных структур с наследованием.

Примерные вопросы к зачету

Расскажите про паттерн одиночка в ООП.

Для чего существуют модификаторы доступа в ООП.

Для чего нужна абстракция в ООП?

Как реализуется конструктор в питоне?

Какие базовые декораторы питона вы знаете и можно ли написать свой декоратор?

Расскажите про паттерн наблюдатель в ООП.

Какие парадигмы в программировании вы знаете кроме ООП и на чем они основываются?

Можно ли в питоне реализовать интерфейс?

Что вы можете сказать про self, cls?

public, private, protected — дорожку к какой парадигме в ООП они открывают и как?

Что такое композиция и в чем ее преимущество перед наследованием?

Алмазная проблема наследования — в чем она заключается и как ее решили?

Что такое агрегация и в чем ее преимущество перед наследованием?

Возможно ли множественное наследование в питоне от обычных классов? Есть ли в этом какая-то проблема?

Что такое перегрузка методов и как она реализована в питоне?

Что такое классы и объекты в ООП?

Разница между методами экземпляра и класса?

Перечислите магические методы в питоне и что они делают.

Разница между экземпляром и объектом класса?

Что такое super() и зачем он нужен?

Как проверить принадлежность объекта к классу?

Разница между new и init?

Что такое полиморфизм в Python?

Что произойдёт, если метод с одинаковым именем определён в родительском и дочернем классах?

Что такое абстрактный класс?

Что такое SOLID-принципы? Назовите и опишите каждый.

В чём плюсы и минусы ООП по сравнению с процедурным подходом?

Как работает **init**()?

Как реализовать шаблон проектирования Наблюдатель на Python?

Что такое композиция в ООП и когда её лучше использовать?

Что такое лямбда-выражения в Python и в чем их основная функция?

Как контролировать доступ к атрибутам при помощи @property?

Чем отличается перегрузка метода от переопределения?

Можно ли создать класс без атрибутов и методов? В чём смысл?

Чем плоха прямая работа с атрибутами экземпляра?

Что произойдёт, если в дочернем классе не вызвать super().**init**()?

Как реализовать поведение «по контракту» в Python?

Что будет, если попытаться переопределить приватный атрибут в подклассе?

Как скрыть атрибут внутри класса?

Чем отличается метод от функции в Python?

Что такое метод **init**() и когда он вызывается?

Можно ли изменить поведение родительского метода в дочернем классе?

Как запретить создание новых атрибутов у объекта?

Как определить, является ли класс наследником другого класса, используя issubclass()?

В чём разница между полиморфизмом и инкапсуляцией?

Как можно передать параметры в класс, но без явного конструктора?

Для чего существуют магические методы?

В чем разница между композицией и агрегацией?

Можно ли у класса и экземпляра быть атрибутами с одинаковыми именами?

Что будет, если в классе не определить init, но при этом передать параметры при создании объекта?

Можно ли создать объект без вызова конструктора?

Что делает staticmethod и как он связан с ООП?

Что произойдёт, если в дочернем классе забыть реализовать метод, определённый как абстрактный?

Что произойдёт, если создать класс без тела и попытаться создать его экземпляр?

Можно ли снаружи изменить значение атрибута, помеченного как приватный?

Как создать полиморфный метод, который работает с числами, строками и списками?

Чем отличаются методы, определённые через def, @staticmethod и @classmethod?

Как сделать, чтобы у дочернего класса не было доступа к методам родителя?

Можно ли реализовать полиморфизм через *args и **kwargs?

Можно ли передать класс как аргумент функции?

Можно ли создать класс внутри другого класса? Зачем это может быть нужно?

Как сделать атрибут доступным только для чтения, но видимым извне?

Можно ли обращаться к атрибуту класса через экземпляр, и что будет, если изменить его значение?

Можно ли изменить атрибуты класса через его экземпляр?

Какие проблемы могут возникнуть при неправильном использовании множественного наследования?

Можно ли изменить атрибуты класса через его экземпляр?

Что такое «объект» в ООП и как его можно создать?

Можно ли использовать методы класса без создания экземпляра?

Что такое «агрегация» в ООП и как она отличается от «композиции»?

Как можно проверить, существует ли атрибут у объекта?

В чём разница между абстрактным методом и обычным методом в ООП?