Описание технологии проведения экзамена:

- формат проведения экзамена: устный экзамен в **формате беседы** по вопросам экзаменационного билета и решения практической задачи,
- порядок формирования экзаменационного билета: 1-й вопрос – с 1 по 5 вопрос из перечня вопросов к экзамену, 2-й вопрос – с 6 по 32 вопрос, вопросы и задача выбирается из перечня случайным образом.
- требования к ответу: изложение должно быть логичным, придерживаться четкого плана, для пояснения привести примеры.

Примерный перечень вопросов к экзамену:

1. Жизненный цикл программного продукта: определение, необходимость его изучения.
2. Основные процессы жизненного цикла программного продукта.
3. Каскадная модель жизненного цикла программного продукта. Возможность применения в современных технологиях.
4. Спиральная модель жизненного цикла программного продукта. Возможность применения в современных технологиях.
5. Модели SCRUM и Kanban.
6. Основные элементы объектно-ориентированного подхода (ОО-анализ, ОО-проектирование, модель предметной области).
7. Определение класса.
8. Атрибуты и методы класса.
9. Конструктор класса.
10. Деструктор класса.
11. Определение классов в модулях.
12. Понятие инкапсуляции. Использование специальных методов доступа.
13. Понятие инкапсуляции. Применение аннотаций свойств.
14. Отношения между классами. Зависимость.
15. Отношения между классами. Композиция и агрегация.
16. Отношения между классами. Наследование.
17. Определение (перегрузка) операторов.
18. Понятие и реализация абстрактного класса.
19. Реализация наследования абстрактного класса.
20. Реализация множественного наследования. Применение Mixins.
21. Порядок разрешения методов (Method Resolution Order).
22. Декораторы и их применение.
23. Метаклассы и их применение.
24. Обработка исключений.
25. Использование иерархии классов исключений.
26. Реализация многопоточности.
27. Реализация асинхронного программирования.
28. Создание GUI в ООП стиле.
29. Применение ООП при создании компонентов GUI.
30. Создание приложений, взаимодействующих с базой данных SQLite.
31. Применение ORM (peewee).
32. Применение ORM (SQLAlchemy).

Примерный перечень практических заданий

Задача 1. Создайте класс с именем Time, содержащий три атрибута, предназначенные для хранения часов, минут и секунд. Конструктор класса должен определить значения по умолчанию для некоторых аргументов. Создайте метод класса, который будет выводить значения полей на экран в формате 11:59:59, и метод, складывающий значения двух объектов типа Time. Выполните тестирование – создайте два инициализированных объекта и один неинициализированный объект. Затем сложите два инициализированных значения, а результат присвойте третьему объекту и выведите его значение на экран.

Задача 2. Создайте класс Employee. Класс должен включать атрибуты для хранения номера сотрудника и величины его оклада. Методы класса должны позволять пользователю вводить и отображать эти данные класса. Напишите функцию, которая запросит пользователя ввести данные для трех сотрудников и выведет полученную информацию на экран.

Задача 3. Создайте класс Publication, в котором хранятся название и цена книги. Реализуйте наследование этого класса – два класса: первый (бумажная книга) содержит информацию о количестве страниц в книге, второй (аудиокнига) время записи книги в минутах. В каждом из этих трех классов должен быть метод getdata(), с помощью которого можно получать данные от пользователя, и putdata(), предназначенный для вывода этих данных. Напишите функцию main() для проверки классов, создайте их объекты в программе и запросите у пользователя данные, выведете их на экран.

Задача 4. Создайте класс, одно из полей которого хранит «порядковый номер» объекта, то есть для первого созданного объекта значение этого поля равно 1, для второго созданного объекта значение равно 2 и т. д. В класс включите метод, который будет выводить на экран порядковый номер объекта. Напишите функцию main(), в которой создайте три объекта, и реализуйте вывод объектом своего порядкового номера, например: Мой порядковый номер: 2.

Задача 5. Исходные данные: Телефонная компания хранит данные о своих клиентах: имя и баланс счета. Выполняется пополнение баланса и обрабатываются звонки клиента в зависимости от типа звонка: «городской» (5 руб./мин.) и «мобильный» (1 руб./мин.), а также количества минут разговора.

Проектное решение (вариант):

Для учета клиентов используется класс `Customer` (Клиент), содержащий атрибуты:

- поле `name`: имя клиента (чтение/запись);
- поле `balance`: баланс счета клиента (только чтение);
- метод `record_payment()`: выполняет пополнение баланса;
- метод `record_call()`: выполняет обработку звонка клиента в зависимости от:
  - типа звонка: «городской» (5 руб./мин.) и «мобильный» (1 руб./мин.);
  - количества минут разговора.

Требуется: Заказчик желает расширить возможности компании: добавить клиенту возможность выбора тарифного плана, в каждом из которых есть тип звонка («городской» и «мобильный»):

- `Повременный`: «городской» (5 руб./мин.) и «мобильный» (1 руб./мин.);
- `После10МинутВ2РазаДешевле`: после 10 минут звонка на городской номер каждая вторая минута бесплатно; в остальном как `Повременный`;
- `ПлатиМеньшеДо5Минут`: до 5 минут разговора в 2 раза дешевле тарифа `Повременный`, после - в 2 раза дороже.

Предложите решение в виде прототипа (базовая реализация диаграммы классов, можно подробно не раскрывать содержимое класса/классов), реализующее желание заказчика.