ДИСЦИПЛИНА Разработка приложений на языке Котлин
(полное наименование дисциплины без сокращений)

ИНСТИТУТ информационных технологий

КАФЕДРА информационных технологий в атомной энергетике
(полное наименование кафедры)

ВИД УЧЕБНОГО МАТЕРИАЛА Практическая работа
(в соответствии с пп 1-11)

ПРЕПОДАВАТЕЛЬ Золотухин Святослав Александрович
(фамилия, имя, отчество)

СЕМЕСТР 5 семестр 2025 – 2026 учебный год
(указать семестр обучения, учебный год)

ЗАДАНИЕ ДЛЯ 7 ПРАКТИЧЕСКОЙ РАБОТЫ ПО ДИСЦИПЛИНЕ «РАЗРАБОТКА ПРИЛОЖЕНИЙ НА ЯЗЫКЕ КОТЛИН»

Задание:

Добавить тестовые данные для проверки разработанного функционала.

Вариант 1. Библиотека цифровых ресурсов

Спроектируйте и реализуйте систему классов для представления цифровых ресурсов в библиотеке.

Часть 1.

1. Создайте абстрактный класс *DigitalResource.*
2. Объявите абстрактное свойство *title* (название) с типом *String*.
3. Объявите абстрактный метод *getDescription()*, который возвращает строку с описанием ресурса.
4. Добавьте общее для всех ресурсов свойство *fileSizeInBytes* (размер файла в байтах). Это свойство должно быть инициализировано через конструктор базового класса и не должно быть доступно для переопределения в дочерних классах.
5. Добавьте свойство *isDownloadable* (можно ли скачать) с геттером по умолчанию, возвращающим *true*. Сеттер этого свойства должен быть закрыт для переопределения.

Часть 2.

1. Создайте data-класс *ResourceMetadata*, который будет хранить информацию об авторе и дате публикации.
2. Добавьте в класс *DigitalResource* свойство *metadata* типа *ResourceMetadata?* (может быть null), которое должно быть проинициализировано значением по умолчанию null.

Часть 3.

1. Создайте класс *EBook*, наследующийся от *DigitalResource*.
2. Переопределите свойство *title*, сделав его *override val*.

3. Реализуйте метод *getDescription()*. Пусть он возвращает строку: «Электронная книга: [title]».
4. Добавьте специфическое свойство *pageCount* (количество страниц).
5. **Используйте различные конструкторы.** Создайте основной конструктор для *EBook*, принимающий *title, fileSizeInBytes и pageCount*. Создайте вторичный конструктор, который принимает *title и pageCount, а fileSizeInBytes* вычисляет приблизительно как pageCount * 1024 (условно, 1Кб на страницу).

Часть 4.

1. Создайте класс *AudioBook*, наследующийся от *DigitalResource*.
2. Переопределите геттер свойства *isDownloadable*. Пусть он всегда возвращает *false* (аудиокниги можно только слушать онлайн).
3. Реализуйте метод *getDescription()*. Пусть он возвращает строку: «Аудиокнига: [title]. Время прослушивания: [duration] мин.»
4. Добавьте специфическое свойство *duration* (продолжительность в минутах).

Вариант 2. Система заказов в кафе

Реализуйте иерархию классов для системы формирования заказов в кафе.

Часть 1.

1. Создайте абстрактный класс *MenuItem*.
2. Объявите абстрактное свойство *name* (название).
3. Объявите абстрактное свойство *basePrice* (базовая цена).
4. Объявите абстрактный метод *calculateFinalPrice()*, который будет вычислять итоговую цену с учетом всех надбавок (например, за размер порции).
5. Свойство *id* (уникальный идентификатор) должно быть задано в базовом классе как *val id: String = UUID.randomUUID().toString()*. Запретите переопределение этого свойства.

Часть 2.

1. Создайте data-класс *Ingredient* с полями *name* и *isAllergen* (является ли аллергеном).

Часть 3.

1. Создайте класс *Drink*, наследующийся от *MenuItem*.
2. Переопределите свойства *name* и *basePrice*.
3. Добавьте свойство *size* (тип Size). Создайте *enum class Size { SMALL, MEDIUM, LARGE }*.
4. Переопределите метод *calculateFinalPrice()*. Логика: *SMALL - basePrice * 1.0, MEDIUM - basePrice * 1.5, LARGE - basePrice * 2.0*.
5. Запретите дальнейшее переопределение метода *calculateFinalPrice()* в классах-потомках (если они будут).

Часть 4.

1. Создайте класс *Food*, наследующийся от *MenuItem*.
2. Переопределите свойства *name* и *basePrice*.
3. Добавьте свойство *ingredients*: *List<Ingredient>*, содержащее список ингредиентов.
4. Добавьте свойство *isVegetarian*. Реализуйте его кастомный геттер, который проверяет, что все ингредиенты не являются аллергенами (просто пример логики, в реальности это не так). Геттер должен возвращать *true*, если в списке *ingredients* нет ни одного ингредиента с *isAllergen = true*.
5. Переопределите метод *calculateFinalPrice()*. Итоговая цена равна *basePrice*.

Требования к отчету:

Титульный лист, оглавление, текст задачи, выполненные задания (краткое описание кода реализации каждого задания с указанием листинга кода и скриншотов работы каждой программы), вывод (что было сделано в ходе выполнения работы), список использованных источников.

Оформление работ обязательно должно отвечать требованиям СМКО МИРЭА.

При защите работы необходимо ответить на несколько контрольных вопросов.

Литература:

1) Лекционный материал
2) https://kotlinlang.org/docs/home.html