**Модуль №5. Задание №3**

Решение задач присылать ссылкой на github репозиторий, в Discord чат «с-домашняя-работа».

Структуры данных

**Задача №1. Реализация очереди для управления заказами в ресторане**

Создайте класс **OrderQueue**, который будет использовать очередь для управления заказами в ресторане. Данный класс имеет следующий протокол работы:

* **place\_order(order)** - помещает заказ в очередь.
* **serve\_order()** - удаляет и возвращает первый заказ из очереди для обслуживания.
* **peek\_next\_order()** - возвращает первый заказ из очереди без его удаления.
* **is\_empty()** - проверяет, пуста ли очередь заказов.
* **waiting\_orders\_count()** - возвращает количество заказов, ожидающих обслуживания.

Сущность заказ будет описана классом **Order**, который состоит атрибутов: id – идентификатор заказа, price – цена за весь заказ. Все атрибуты имеют закрытый уровень доступа и инкапсулируются геттерами.

Сформируйте тест-план, выделив виды тестов и опишите тест-кейсы. Сформируйте тест-функции на каждую операцию класса.

**Задача №2. Разработка системы обработки технических заявок**

Создайте класс **SupportQueue** для управления техническими заявками в IT-компании. Каждая заявка имеет уровень приоритета и описание проблемы. Сущность заявки представлена классом **Request**, который состоит из атрибутов: id – идентификатор заявки, description – описание проблемы, priority – уровень приоритета.

Класс SupportQueue состоит из следующих операций:

* **submit\_request(request, request\_priority)** - добавляет новую заявку в очередь.
* **resolve\_request()** - удаляет и возвращает заявку с наивысшим приоритетом из очереди.
* **peek\_next\_request()** - возвращает заявку с наивысшим приоритетом без ее удаления из очереди.
* **is\_empty()** - проверяет, пуста ли очередь заявок.

Сформируйте тест-план, выделив виды тестов и опишите тест-кейсы. Сформируйте тест-функции на каждую операцию класса.

**Задача №3. Разработка класса для двунаправленного списка**

Разработайте сущность **DoubleLinkedList** для работы с структурой данных двунаправленный список. Сущность должна реализовать следующий протокол:

* **add\_last(value)** - добавление нового узла с заданным значением в конец списка.
* **add\_first(value)** - добавление нового узла с заданным значением в начало списка.
* **insert(value, position)** - вставка нового узла на указанную позицию.
* **delete(position)** - удаление узла на указанной позиции.
* **search(value)** - поиск узла по значению и возврат его позиции.

Сформируйте тест-план, выделив виды тестов и опишите тест-кейсы. Сформируйте тест-функции на каждую операцию класса.