Домашнее задание №1. ООП

Данное задание является стартовым для курса и включает несколько целей:

1. Отработка навыка решения типовых объектно-ориентированных задач с использованием языка Java. В ходе выполнения работы необходимо разработать группу классов, которые реализуют прописанные в задании требования.
2. Отработать навык формирования исполняемого приложения на языке Java
3. Отработать навык упаковки Java приложений в исполняемый архив.

Задание состоит из набора требований, каждое из которых оценивается в определенное количество баллов.

Для первых трех заданий оценка производится путем код ревью представленного решения, на предмет его соответствия объектно-ориентированному подходу. В случае выполнения решения процедурным подходом оценка снижается, или решение отправляется на переделку. Для проверки соответствия ООП подходу, к каждой части задания приложены критерии оценивания.

Следующие два задания направлены на отработку навыка формирования исполняемого приложения. Для этого необходимо реализовать инфраструктуру с **main** методом, который будет исполняем и обладать консольным интерфейсом.

Последнее задание является бонусным и может быть выполнено при наличии знаний о модульном тестировании с использованием Java.

Итого, за работу можно получить 6 основных и один бонусный балл.

Часть 1. Разработка базового класса (1 балл)

Необходимо разработать класс **Студент**. Данный класс должен отвечать следующим требованиям:

1. Имеет имя (строка)
2. Имеет список оценок (числовой)
3. Имя можно изменить или узнать в любой момент времени
4. Оценки можно добавлять, удалять и просматривать
5. Два **Студента** равны если у них одинаковые имена и список оценок
6. Текстовое представление студента имеет вид: “Имя: [оценка1, оценка2,…,оценкаN]”
7. Студент может быть создан двумя способами:
    a. с указанием имени
    b. с указанием имени и списка оценок.

Критерии оценивания:

- Выполнены требования по наличию полей\методов\конструкторов
- Состояние объекта защищено от внесения некорректных значений

Часть 2. Добавление условий. (1 балл)

Необходимо внести следующие изменения в базовый код **Студента**:

1. Тип оценки может быть не только числом, но вообще любым типом данных, например строкой или датой. Тип оценки указывается при создании объекта.
2. При создании объекта **Студента** ему может быть задано указание на то, что является корректной оценкой, а что нет (например четные числа в диапазоне от 1 до миллиона, или слова “зачет” и “незачет”). Если признак корректности не задан, значит все значения данного типа данных являются корректными оценками.
3. При попытке добавить **Студенту** некорректную оценку должно выбрасываться исключение.
4. Не должно быть способа добавить некорректную оценку.

Критерии оценивания:

- Изменение типа данных оценки не требует изменения класса **Студента**
- Признак корректности может быть любым мыслимым условием, и его изменение не требует изменения класса **Студента**

Часть 3. Отмена. (2 балла)

Необходимо внести следующие изменения в код **Студент** полученный в части 2:

Добавьте Студенту метод отмены последнего действия. Поскольку у Студента можно изменить имя, добавить или удалить оценку, то отмена действия будет приводить к удалению или добавлению оценки либо возвращению предыдущей версии имени. Метод отмены можно вызывать до тех пор, пока по очереди не отменятся все действия и Студент не вернется к начальному состоянию.

Критерии оценивания:

- Метод отмены не требует параметров
- Появление у класса дополнительных полей не потребует внесения изменений никуда, кроме как в методы получения и изменения значений добавленных полей. При этом у них также должна быть возможность участвовать в “отмене действия”.

Часть 4. Главный метод (1 балл)

Реализуйте консольный интерфейс для Java приложения, который будет предлагать следующий набор действий:

1. Создание объекта класса **Студент** с указанным именем. При повторном вызове действия, ранее созданные **Студент** идет в мусор, и в дальнейшем работа производится с этим объектом.

2. Добавление одной оценки объекту ранее созданного **Студента**.
3. Удаление последней оценки с указанным значением у **Студента**. Если такой оценки нет – ничего не происходит.
4. Печать текстового описания **Студента** на экране.

Критерии оценивания:

- Приложение запускается
- Ни один из пунктов меню не приводит к возникновению ошибок
- Действия выполняют именно то что надо.
- Желательно приложить скрипт для запуска указанных действий, демонстрирующий их выполнение.
- Если выполнена часть 5, то проверка выполняется с её использованием

Пример выполнения:

1. Создан Студент Иван
2. Ивану добавлены оценки 5, 3 ,4
3. Удалена оценка 3
4. Студент выведен на экран. На экране окажется текст: “Иван: [5, 4]”

Часть 5. Архив (1 балл)

Упакуйте ранее созданное приложение в запускаемый JAR архив.

Бонусное задание: тесты (1 балл)

Предоставьте в репозитории набор Junit тестов, которые тестируют корректность реализации кода для частей 1-3 задания.

Критерии оценивания:

1. Тесты запускаются Maven или Gradle скриптом
2. Code Coverage 100% для методов класса **Студента**