Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №3-4

Вариант №8000088.2

Выполнил:
Решетников Сергей Евгеньевич
Группа P3108
Проверил:

Санкт-Петербург 2024

Оглавление

1. Задание

Описание предметной области, по которой должна быть построена объектная модель:

> Отряхнув от земли один клубень, Скуперфильд откусил кусочек и попробовал его разжевать. Сырой картофель показался ему страшно невкусным, даже противным. Сообразив, однако, что никто не стал бы выращивать совершенно бесполезных плодов, он сунул вытащенные из земли полдесятка картофелин в карман пиджака и отправился дальше. Шагать по рыхлой земле, беспрерывно путаясь ногами в картофельной ботве, было очень утомительно. Скуперфильд на все лады проклинал коротышек, вздумавших, словно ему назло, взрыхлить вокруг землю и насадить на его пути все эти кустики. Как и следовало ожидать, ему все же удалось в конце концов добраться до края картофельного поля. Выбравшись на твердую почву, Скуперфильд облегченно вздохнул и в тот же момент ощутил доносившийся откуда-то запах дыма. От этого запаха на него словно повеяло теплом и домашним уютом.

Этапы выполнения работы:

1. Получить вариант
2. Нарисовать UML-диаграмму, представляющую классы и интерфейсы объектной модели и их взаимосвязи;
3. Придумать сценарий, содержащий действия персонажей, аналогичные приведенным в исходном тексте;
4. Согласовать диаграмму классов и сценарий с преподавателем;
5. Написать программу на языке Java, реализующую разработанные объектную модель и сценарий взаимодействия и изменения состояния объектов. При запуске программа должна проигрывать сценарий и выводить в стандартный вывод текст, отражающий изменение состояния объектов, приблизительно напоминающий исходный текст полученного отрывка.
6. Продемонстрировать выполнение программы на сервере `helios`.
7. Ответить на контрольные вопросы и выполнить дополнительное задание.

Текст, выводящийся в результате выполнения программы не обязан дословно повторять текст, полученный в исходном задании. Также не обязательно реализовывать грамматическое согласование форм и падежей слов выводимого текста.

Стоит отметить, что цель разработки объектной модели состоит не в выводе текста, а в эмуляции объектов предметной области, а именно их состояния (поля) и поведения (методы). Методы в разработанных классах должны изменять состояние объектов, а выводимый текст должен являться побочным эффектом, отражающим эти изменения.

Требования к объектной модели, сценарию и программе:

1. В модели должны быть представлены основные персонажи и предметы, описанные в исходном тексте. Они должны иметь необходимые атрибуты и характеристики (состояние) и уметь выполнять свойственные им действия (поведение), а также должны образовывать корректную иерархию наследования классов.
2. Объектная модель должна реализовывать основные принципе ООП - инкапсуляцию, наследование и полиморфизм. Модель должна соответствовать принципам SOLID, быть расширяемой без глобального изменения структуры модели.
3. Сценарий должен быть вариативным, то есть при изменении начальных характеристик персонажей, предметов или окружающей среды, их действия могут изменяться и отклоняться от базового сценария, приведенного в исходном тексте. Кроме того, сценарий должен поддерживать элементы случайности (при генерации персонажей, при задании исходного состояния, при выполнении методов).
4. Объектная модель должна содержать как минимум один корректно использованный элемент каждого типа из списка:
    - абстрактный класс как минимум с одним абстрактным методом;
    - интерфейс;
    - перечисление (enum);
    - запись (record);
    - массив или ArrayList для хранения однотипных объектов;
    - проверяемое исключение.
5. В созданных классах основных персонажей и предметов должны быть корректно переопределены методы `equals()`, `hashCode()` и `toString()`. Для классов-исключений необходимо переопределить метод `getMessage()`.
6. Созданные в программе классы-исключения должны быть использованы и обработаны. Кроме того, должно быть использовано и обработано хотя бы одно unchecked исключение (можно свое, можно из стандартной библиотеки).
7. При необходимости можно добавить внутренние, локальные и анонимные классы.

2. UML-диаграмма

Доступна по ссылке — https://github.com/NF-coder/ITMO_repo/tree/main/programming/sem1/lab3/uml.png

3. Исходный код программы

Исходники доступны по ссылке - https://github.com/NF-coder/ITMO_repo/tree/main/programming/sem1/lab3

4. Результат работы программы

https://github.com/NF-coder/ITMO_repo/tree/main/programming/sem1/lab3/out.txt

5. Вывод

Во время выполнения лабораторной работы я изучил принципы SOLID, научился использовать интерфейсы, enum’ы, абстрактные классы и record’ы. Помимо этого я более детально изучил ООП