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

Факультет программной инженерии и компьютерной техники

Лабораторная работа по программированию №3-4

Вариант №31056

Выполнил:
Студент группы P3106
Мельник Фёдор Александрович
Проверил:
Вербовой. А. А.,
Преподаватель-практик ФПиКТ

Санкт-Петербург, 2024

Задание

В соответствии с выданным вариантом на основе предложенного текстового отрывка из литературного произведения создать объектную модель реального или воображаемого мира, описываемого данным текстом. Должны быть выделены основные персонажи и предметы со свойственным им состоянием и поведением. На основе модели написать программу на языке Java.

Введите вариант: 31056

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

> Все опять громко захохотали. Незнайка подошел к раковине и стал умываться под краном. А коротышки заспорили между собой. Одни утверждали, что Незнайка нарочно придумывает разные небылицы, чтоб сбить с толку полицию; другие говорили, что он попросту дурачок и болтает, что придет в голову; третьи решили, что он сумасшедший. Тот, который был без рубахи, уверял всех, что Незнайка, должно быть, свихнулся с ума, начитавшись книжек, а в книжках на самом деле сказано, что за наружной Луной есть какие-то огромные планеты и звезды, на которых тоже якобы живут коротышки. Вот он и вообразил, наверно, что прилетел к нам с такой планеты. Сумасшедшие всегда воображают себя какими-нибудь великими личностями, знаменитостями или отважными путешественниками.

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.

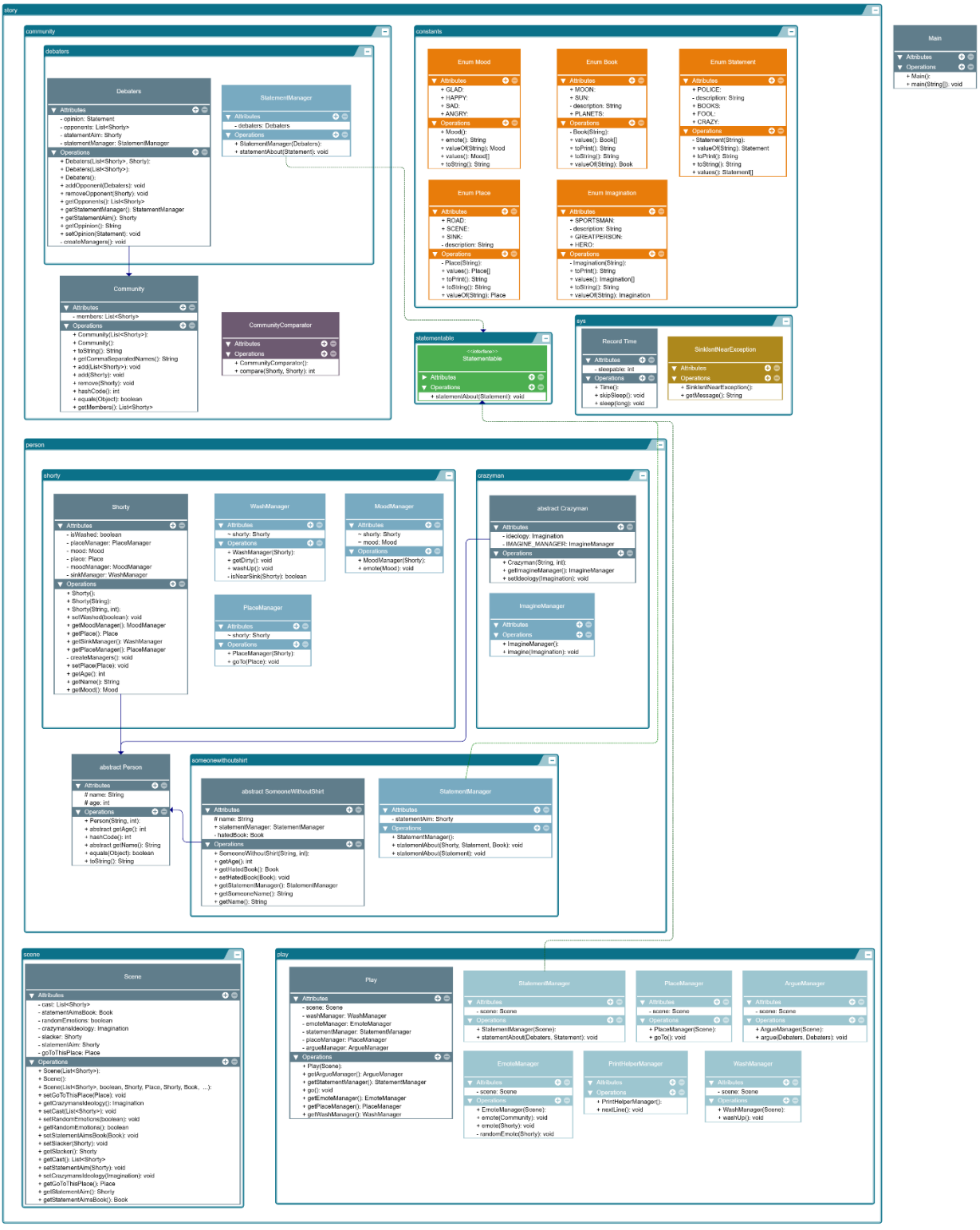
Текст, выводящийся в результате выполнения программы не обязан дословно повторять текст, полученный в исходном задании. Также не обязательно реализовывать грамматическое согласование форм и падежей слов выводимого текста.

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.

Диаграмма классов объектной модели



В хорошем качестве: https://github.com/ldpst/itmo/blob/main/sem-1-2_prog/labs/lab3-4/diagrams/UML_prog_lab3-4.png

Исходный код программы

https://github.com/ldpst/itmo/tree/main/sem-1-2_prog/labs/lab3-4/src

Результат работы программы

Незнайка захохотал.
Знайка захохотал.
Кнопочка захохотал.
Пилюлькин захохотал.
Пончик захохотал.
Ромашка захохотал.
Мушка захохотал.

Незнайка подошёл к раковина.

Незнайка пытается умыться:
Незнайка успешно умылся.
Незнайка помытый.

Пилюлькин, Ромашка спорят с Знайка, Кнопочка.
Знайка, Кнопочка спорят с Пилюлькин, Ромашка.
Мушка, Пончик спорят с Знайка, Кнопочка.
Знайка, Кнопочка спорят с Мушка, Пончик.
Пилюлькин, Ромашка спорят с Мушка, Пончик.
Мушка, Пончик спорят с Пилюлькин, Ромашка.

Пилюлькин, Ромашка утверждают: Незнайка нарочно придумывает разные небылицы, чтоб сбить с толку полицию.
Знайка, Кнопочка утверждают: Незнайка попросту дурачок и болтает, что придет в голову.
Мушка, Пончик утверждают: Незнайка сумасшедший.

Тот, который без рубахи, утверждает: Незнайка, должно быть, свихнулся с ума, начитавшись книжек. А в книжках на самом деле сказано: за наружной Луной есть какие-то огромные планеты и звезды, на которых тоже якобы живут коротышки. Вот он и вообразил, наверно, что прилетел к нам с такой планеты.

Сумасшедшие воображают себя какими-нибудь великими личностями, знаменитостями или отважными путешественниками.

Вывод

При выполнении лабораторной работы я познакомился с принципами SOLID. Реализовывал собственные интерфейсы, enum’ы, record’ы и абстрактные классы, а также их взаимодействие друг с другом. В ходе работы я получил важнейший опыт по изготовлению спагетти и его распутыванию. Данная работа сильно прокачала мои навыки в написании программ, в которых используется ООП.