

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

Факультет программной инженерии и компьютерной техники

Лабораторная работа по программированию №3-4

Вариант №31056.13

Выполнил:

Студент группы Р3106

Мельник Фёдор Александрович

Проверил:

Вербовой. А. А.,

Преподаватель-практик ФПиКТ

Санкт-Петербург, 2024

Оглавление

Задание	3
Диаграмма классов объектной модели	4
Исходный код программы	5
Результат работы программы	6
Вывод	7

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:

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

Помещение было без окон и освещалось одной-единственной электрической лампочкой, висевшей высоко под потолком. Лампочка была тусклая и светила, как говорится, только себе под нос. Как только Незнайка попал в каталажку и дверь за ним захлопнулась, он принялся протирать руками глаза, пытаясь хоть что-нибудь разглядеть в полутьме. Толку из этого вышло мало: он лишь размазал по лицу черную краску, которой были испачканы его руки. Увидев новоприбывшего, несколько самых любопытных коротышек соскочили со своих полок и подбежали к нему. Незнайка в испуге попытался и, прижавшись спиной к двери, приготовился защищаться. Разглядев его измазанную физиономию, коротышки невольно рассмеялись. Незнайка понял, что бояться не надо, и его лицо тоже расплылось в улыбку. Громкий смех заглушил его слова. Все хорошо знали, что фертинги -- это не что иное, как деньги, поэтому Незнайкины слова были сочтены за остроумную шутку. Он был без рубашки. Как раз в тот момент, когда Незнайка вошел, он зашивал на рубашке дырку, и теперь так и стоял с иголкой в руке. 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 рубашки, 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. В это время Незнайка кончил умываться и спросил: Незнайка подсел к коротышкам, которые грелись у печи. Стриженный тоже сел рядом. Стриженный достал из кармана несколько медных монеток. С этими словами он стащил с Незнайки его голубую шляпу и сунул в руку монетки. Вихрастый взял у Незнайки монетки. Десять сантиков он незаметно сунул себе в карман, а пять сантиков зажал в кулаке и, подойдя к двери, негромко стукнул три раза. Звякнул замок. Дверь приоткрылась, и в нее заглянул уже знакомый нам полицейский Дриль.

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, выводимый 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 принципы 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

https://github.com/ldpst/prog_lab3-4_v3/blob/main/prog_lab3-4_v3_uml.png

Исходный код программы

https://github.com/ldpst/prog_lab3-4_v3/tree/main/src

Результат работы программы

В помещении каталажка 0 окон.

Помещение каталажка освещается одной лампочкой.

Лампочка висит высоко.

Лампочка светит тускло.

Незнайка подошёл к каталажка.

Дверь каталажка захлопнулась

Незнайка загрустил.

Знайка злится.

Кнопочка стал счастлив.

Пилюлькин захохотал.

Пончик захохотал.

Ромашка загрустил.

Мушка злится.

Незнайка подошёл к раковина.

Незнайка пытается умыться:

Незнайка успешно умылся.

Незнайка помытый.

Пилюлькин, Кнопочка спорят с Знайка, Ромашка.

Знайка, Ромашка спорят с Пилюлькин, Кнопочка.

Пончик, Мушка спорят с Знайка, Ромашка.

Знайка, Ромашка спорят с Пончик, Мушка.

Пилюлькин, Кнопочка спорят с Пончик, Мушка.

Пончик, Мушка спорят с Пилюлькин, Кнопочка.

Пилюлькин, Кнопочка утверждают: Незнайка нарочно придумывает разные небылицы, чтоб сбить с толку полицию.

Знайка, Ромашка утверждают: Незнайка попросту дурачок и болтает, что придет в голову.

Пончик, Мушка утверждают: Незнайка сумасшедший.

Тот, который без рубахи, утверждает: Незнайка, должно быть, свихнулся с ума, начитавшись книжек. А в книжках на самом деле сказано: за нашими планетами есть какие-то другие 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.