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

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №3

По программированию

Вариант 23468

Выполнил:

Ларионов Владислав Васильевич

Группа P3109

Проверил:

Мустафаева Айнур Вугар Кызы

Санкт-Петербург 2024

Содержание

[**Задание** 3](#_Toc183973437)

[**Описание предметной области, по которой должна быть построена объектная модель:** 3](#_Toc183973438)

[**Этапы выполнения работы:** 3](#_Toc183973439)

[**Требования к объектной модели, сценарию и программе:** 4](#_Toc183973440)

[**Программа** 5](#_Toc183973441)

[**Результат выполнения:** 5](#_Toc183973442)

[**Диаграмма классов:** 6](#_Toc183973443)

[**Вывод:** 6](#_Toc183973444)

# **Задание**

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

Счастье для бедняка, что к нему подскакал именно Пятница; он у себя на родине привык видеть волков и не боялся их, поэтому он подъехал вплотную к волку и застрелил его, как было описано выше; всякий другой из нас выстрелил бы издали и рисковал бы промахнуться или подстрелить самого проводника. Это могло бы напугать и более смелого человека, чем я, и действительно, весь наш отряд всполошился, когда, вслед за выстрелом, до нас с двух сторон донесся волчий вой, повторяемый горным эхом, так что, казалось, волков было множество; да может и в самом деле их было не так уж мало, чтобы нам совершенно нечего было бояться.

## **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.

# **Программа**

Ссылка на репозиторий, содержащий весь код, jar-файл, весь код, собранный в работающий jar-архив, а также результат выполнения программы и UML-диаграмму:

[ITMO\_VT/prog/lab3\_dir at main · vladlenblch/ITMO\_VT](https://github.com/vladlenblch/ITMO_VT/tree/main/prog/lab3_dir)

## **Результат выполнения:**

горы везде

сугробы повсюду

сани стоят на тропа

пятница подскакивает к бедняк

бедняк подъезжает к волк

вдруг послышался короткий и громкий звук выстрел

бедняк стреляет в волк

волк 2 погибает...

бедняк не боится волк

любой другой стреляет в волк

любой другой промахивается по волк

любой другой боится волк

вдруг послышался долгий и приглушенный звук волчий вой

вдруг послышался долгий и приглушенный звук эхо

отряд не боится волков, ведь их не так много

## **Диаграмма классов:**

Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Автоматически созданное описание

Рис. 1 – диаграмма классов

# **Вывод:**

Во время выполнения данной лабораторной работы я научился:

1. Писать абстрактные классы, интерфейсы, классы типа record и enum.
2. Составлять объектные модели на основе придуманного сценария

Также я повторил свои знания в написании программ на языке программирования Java.