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

Факультет программной инженерии и компьютерной техники

Лабораторная работа

Лабораторная работа по программированию №3

Студент: Трусковский Георгий Александрович (413818)

Номер группы: 3114

Преподаватель: Иманзаде Фахри Рашид Оглы

Санкт-Петербург 2023 год

Оглавление

[Задание 3](#_Toc153276146)

[Лабораторная работа #3 3](#_Toc153276147)

[Реализация Задания 4](#_Toc153276148)

[Вопросы к защите лабораторной работы: 4](#_Toc153276149)

[Ответы на вопросы 5](#_Toc153276150)

[Вывод 6](#_Toc153276151)

# Задание

## Лабораторная работа #3

Введите вариант: 

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

Муми-тролль слез с носа и подсел к ней. Солнце медленно клонилось к закату, расцвечивая мертвую зыбь золотым и зеленым. Все стало золотым или золотистым -- парус, лодка, ее пассажиры. Далеко на горизонте пламенел в лучах заката остров хатифнаттов.

**Программа должна удовлетворять следующим требованиям:**

1. Доработанная модель должна соответствовать [принципам SOLID](https://en.wikipedia.org/wiki/SOLID_(object-oriented_design)).
2. Программа должна содержать как минимум два интерфейса и один абстрактный класс (номенклатура должна быть согласована с преподавателем).
3. В разработанных классах должны быть переопределены методы equals(), toString() и hashCode().
4. Программа должна содержать как минимум один перечисляемый тип (enum).

**Порядок выполнения работы:**

1. Доработать объектную модель приложения.
2. Перерисовать диаграмму классов в соответствии с внесёнными в модель изменениями.
3. Согласовать с преподавателем изменения, внесённые в модель.
4. Модифицировать программу в соответствии с внесёнными в модель изменениями.

**Отчёт по работе должен содержать:**

1. Текст задания.
2. Диаграмма классов объектной модели.
3. Исходный код программы.
4. Результат работы программы.
5. Выводы по работе.

# Реализация Задания

<https://github.com/Sinchi1/ProgLabs/tree/main/lab3>

# Вопросы к защите лабораторной работы:

1. Принципы объектно-ориентированного программирования SOLID и STUPID.
2. Класс Object. Реализация его методов по умолчанию.
3. Особенности реализации наследования в Java. Простое и множественное наследование.
4. Понятие абстрактного класса. Модификатор abstract.
5. Понятие интерфейса. Реализация интерфейсов в Java, методы по умолчанию. Отличия от абстрактных классов.
6. Перечисляемый тип данных (enum) в Java. Особенности реализации и использования.
7. Методы и поля с модификаторами static и final.
8. Перегрузка и переопределение методов. Коварианты возвращаемых типов данных.
9. Элементы функционального программирования в синтаксисе Java. Функциональные интерфейсы, лямбда-выражения. Ссылки на методы.

# Ответы на вопросы

1. SOLID и STUPID:

- SOLID - это аббревиатура, обозначающая пять принципов объектно-ориентированного программирования:

- S - Single responsibility principle (Принцип единственной ответственности)

- O - Open/closed principle (Принцип открытости/закрытости)

- L - Liskov substitution principle (Принцип подстановки Барбары Лисков)

- I - Interface segregation principle (Принцип разделения интерфейса)

- D - Dependency inversion principle (Принцип инверсии зависимостей)

- STUPID, наоборот, описывает противоположные плохие практики, такие как:

- S - Singleton

- T - Tight coupling (жесткая связь)

- U - Untestability (невозможность проведения тестирования)

- P - Premature optimization (преждевременная оптимизация)

- I - Indescriptive naming (неописательное именование)

- D - Duplication (дублирование)

2. Класс Object. Реализация его методов по умолчанию:

- В Java каждый класс является наследником класса Object. Класс Object предоставляет несколько методов по умолчанию, такие как toString(), equals(), hashCode() и finalize(). Эти методы могут быть переопределены в классах-наследниках.

3. Особенности реализации наследования в Java. Простое и множественное наследование:

- В Java поддерживается только одиночное (простое) наследование, то есть класс может наследовать только от одного родительского класса. Это делает иерархию классов более простой и уменьшает сложность взаимодействия между классами. Множественное наследование реализуется через интерфейсы, позволяя классу реализовать несколько интерфейсов.

4. Понятие абстрактного класса. Модификатор abstract:

- Абстрактным классом называется класс, который содержит хотя бы один абстрактный метод (метод без реализации) и отмечен ключевым словом abstract. Абстрактные классы не могут быть инстанциированы напрямую, но могут содержать как абстрактные, так и конкретные методы. Для использования абстрактного класса его нужно расширить (extend) и реализовать все абстрактные методы.

5. Понятие интерфейса. Реализация интерфейсов в Java, методы по умолчанию. Отличия от абстрактных классов:

- Интерфейс в Java представляет собой контракт, описывающий методы, которые должны быть реализованы классом, который имплементирует (implements) интерфейс. Отличие от абстрактных классов в том, что интерфейс не содержит реализации методов. В Java 8 была добавлена поддержка методов по умолчанию в интерфейсы, что позволяет добавлять реализацию метода в интерфейс без необходимости её переопределения в классах, которые его имплементируют

6. Перечисляемый тип данных (enum) в Java. Особенности реализации и использования:

- В Java перечисляемый тип данных (enum) позволяет создавать набор констант. Он удобен для определения переменных, которые представляют фиксированный набор значений. Перечисления могут содержать конструкторы, методы и поля, их также можно использовать в switch-выражениях. Они обеспечивают удобство, безопасность типов и улучшают читаемость кода.

7. Методы и поля с модификаторами static и final:

- static - модификатор, который делает метод или поле общим для всех экземпляров класса. Статические методы и поля могут быть вызваны без создания объекта класса.

- final - модификатор, который делает переменную неизменяемой (для примитивных типов) или ссылку на объект неизменяемой (для объектов). Применяется к методам, чтобы запретить их переопределение в подклассах, а к классам - чтобы запретить наследование.

8. Перегрузка и переопределение методов. Коварианты возвращаемых типов данных:

- Перегрузка методов - это использование одного имени метода для нескольких методов в одном классе, но с различными параметрами.

- Переопределение методов - это процесс создания метода в подклассе с тем же именем, аргументами и возвращаемым типом, что и метод в его суперклассе. Ключевое слово @Override используется для явного указания переопределения метода.

- Коварианты возвращаемых типов данных - в Java 5.0 и выше возвращаемый тип переопределенного метода в подклассе может быть подтипом возвращаемого типа метода в суперклассе.

# Вывод

Узнал основы ООП в виде SOLID, научился аналитически разбивать текст на классы и методы. Научился пользоваться интерфейсами и абстрактными классами.