МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
"МИРЭА - Российский технологический университет"

РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра инструментального и прикладного программного обеспечения

ОТЧЁТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №4
по дисциплине
«Программирование на языке Java»

Тема: Интерфейсы в Java

Выполнил студент группы ИКБО-16-20 Пак С.А.

Принял ассистент кафедры ИиППО Русляков А.А.

Практические работы выполнены «_____» 2021г.

«Зачтено» «_____» 2021г

Москва 2021

СОДЕРЖАНИЕ

ТЕОРЕТИЧЕСКОЕ ВВЕДЕНИЕ

Механизм наследования очень удобен, но он имеет свои ограничения. В частности мы можем наследовать только от одного класса, в отличие, например, от языка C++, где имеется множественное наследование.

В языке Java подобную проблему позволяют решить интерфейсы. Интерфейсы определяют некоторый функционал, не имеющий конкретной реализации, который затем реализуют классы, применяющие эти интерфейсы. И один класс может применить множество интерфейсов.

Чтобы определить интерфейс, используется ключевое слово interface.

```
public interface Printable {
  void print();
}
```

Интерфейс может определять различные методы, которые, так же как и абстрактные методы абстрактных классов не имеют реализации.

Все методы интерфейса не имеют модификаторов доступа, но фактически по умолчанию доступ public, так как цель интерфейса - определение функционала для реализации его классом. Поэтому весь функционал должен быть открыт для реализации.

И также при объявлении интерфейса надо учитывать, что только один интерфейс в файле может иметь тип доступа public. А его название должно совпадать с именем файла. Остальные интерфейсы (если такие имеются в файле java) не должны иметь модификаторов доступа.

Чтобы класс применил интерфейс, надо использовать ключевое слово implements. При этом надо учитывать, что если класс применяет интерфейс, то он должен реализовать все методы интерфейса.

```
class Book implements Printable {
  …
  public void print() {
    // реализация
  }
}
```

РЕШЕНИЕ ЗАДАЧИ

1. Постановка задачи

Вариант: 1

Задание: создать интерфейс Nameable, с методом getName(), возвращающим имя объекта, реализующего интерфейс. Проверить работу для различных объектов (например, можно создать классы, описывающие разные сущности, которые могут иметь имя: планеты, машины, животные и т. д.).

2. Программный код

Файл Nameable.java:

```
package ru.mirea.interfaces;

public interface Nameable {
  String getName();
}
```

Файл Car.java:

```
package ru.mirea.classes;

import ru.mirea.interfaces.*;

public class Car implements Nameable {
  private String name;
  private String color;

  /**
   * Конструктор по умолчанию
   */
  public Car() {
    this.name = "";
    this.color = "";
  }

  /**
   * Конструктор не по умолчанию
   * @param name         название машины
   * @param color        цвет машины
   */
  public Car(String name, String color) {
    this.name = name;
    this.color = color;
  }

  /**
```

```
     * Геттер для поля name
     * @return      название машины
     */
    @Override
    public String getName() {
      return this.name;
    }

    /**
     * Геттер для поля color
     * @return        цвет машины
     */
    public String getColor() {
      return this.color;
    }

    /**
     * Сеттер для поля name
     * @param name      новое название машины
     */
    public void setName(String name) {
      this.name = name;
    }

    /**
     * Сеттер для поля color
     * @param color       цвет машины
     */
    public void setColor(String color) {
      this.color = color;
    }

    /**
     * Объединяет всю информацию об объекте в одну строку
     * @return      строка с информацией об объекте
     */
    @Override
    public String toString() {
      return ("Car {\n"
            + "\tname: " + this.name + "\n"
            + "\tcolor: " + this.color + "\n"
            + "}"
      );
    }
}
```

Файл Phone.java:

```
package ru.mirea.classes;

import ru.mirea.interfaces.*;

public class Phone implements Nameable {
  private String name;
  private String os;
  private double ramSize;

  /**
   * Конструктор по умолчанию
   */
  public Phone() {
    this.name = "";
    this.os = "";
    this.ramSize = 0.0;
  }

  /**
   * Конструктор не по умолчанию
   * @param name          название телефона
   * @param os            операционная система
   * @param ramSize       объём оперативной памяти
   */
  public Phone(String name, String os, double ramSize) {
    this.name = name;
    this.os = os;
    this.ramSize = ramSize;
  }

  /**
   * Геттер для поля name
   * @return    название телефона
   */
  @Override
  public String getName() {
    return this.name;
  }

  /**
   * Геттер для поля os
   * @return      операционная система
   */
  public String getOs() {
    return this.os;
```

```
}

/**
 * Геттер для поля ramSize
 * @return      объём оперативной памяти
 */
public double getRamSize() {
  return this.ramSize;
}

/**
 * Сеттер для поля name
 * @param name      новое название телефона
 */
public void setName(String name) {
  this.name = name;
}

/**
 * Сеттер для поля os
 * @param os    операционная система
 */
public void setOs(String os) {
  this.os = os;
}

/**
 * Сеттер для поля ramSize
 * @param ramSize       новый объём оперативной памяти
 */
public void setRamSize(double ramSize) {
  this.ramSize = ramSize;
}

/**
 * Объединяет всю информацию об объекте в одну строку
 * @return      строка с информацией об объекте
 */
@Override
public String toString() {
  return ("Phone {\n"
    + "\tname: " + this.name + "\n"
    + "\tOS: " + this.os + "\n"
    + "\tramSize: " + this.ramSize + "\n"
    + "}"
  );
}
```

```
}
```

Файл App.java:

```
package ru.mirea;

import java.util.Scanner;

import ru.mirea.classes.*;
import ru.mirea.interfaces.*;

public class App {
  private static final Scanner IN = new Scanner(System.in);

  public static void main(String[] args) {
    Nameable[] goods = new Nameable[4];

    goods[0] = new Car("Toyota Corolla", "red");
    goods[1] = new Car("Subaru", "blue");
    goods[2] = new Phone("Huawei P20 Lite", "Android Oreo", 2.0);
    goods[3] = new Phone("Samsung Galaxy Tab 2", "Android", 1.0);

    System.out.println("Список доступных товаров:");

    for (int i = 0; i < goods.length; ++i) {
      System.out.println((i + 1) + ") " + goods[i].getName());
    }

    System.out.println();
    System.out.println("О каком товаре вы хотите узнать больше?");
    System.out.print("Введите номер товара (1-4): ");

    int goodIndex = IN.nextInt();

    System.out.println();
    System.out.println(goods[goodIndex - 1]);
  }
}
```

3. Вывод программы

Пользователь вводит номер товара в списке, и информация об этом товаре выводится на экран (рис.1, 2).

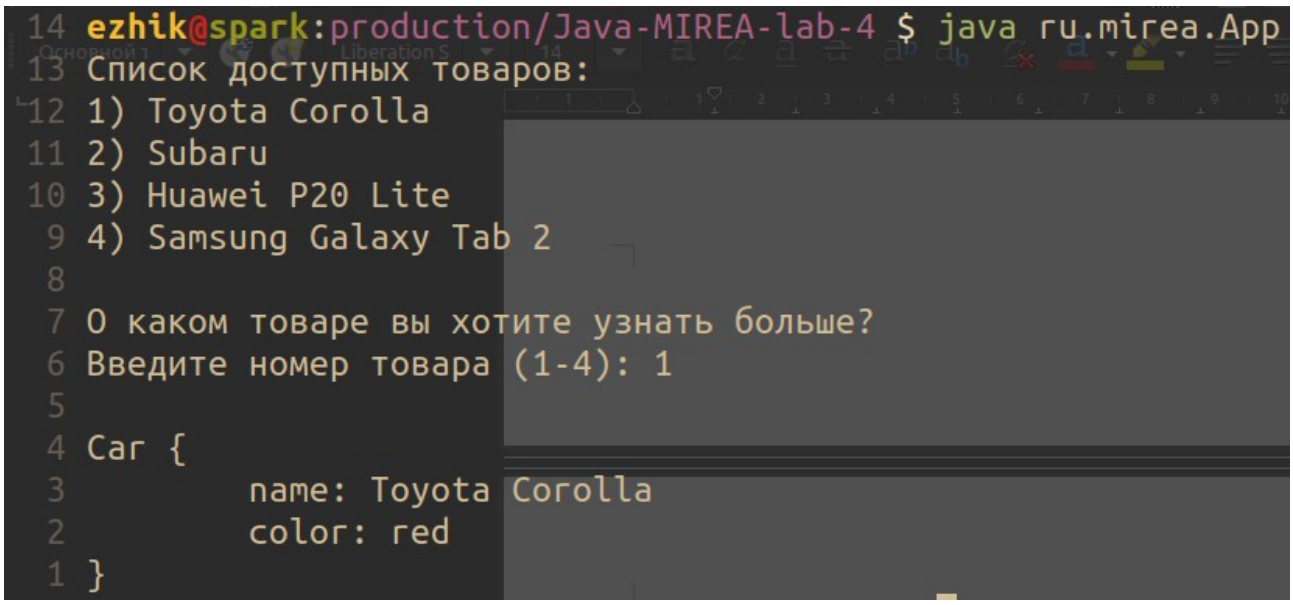

Рис.1 Выходные данные программы ч.1

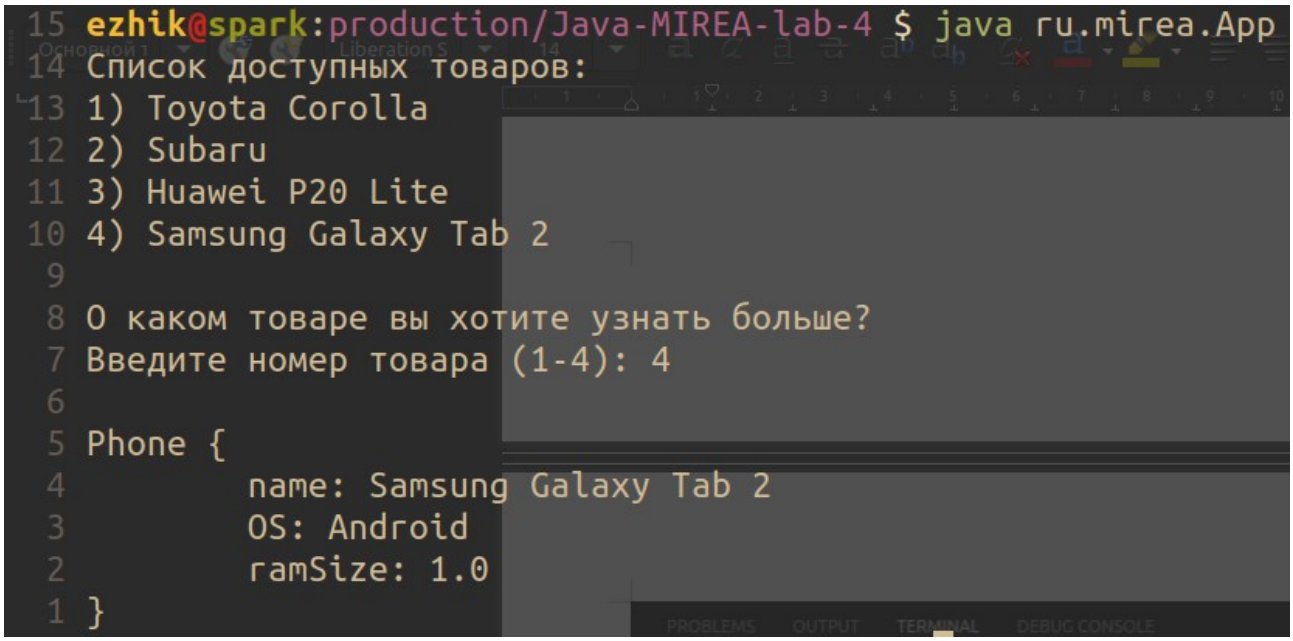

Рис.2 Выходные данные программы ч.2

ВЫВОД

В ходе выполнения изучил понятие интерфейса, научился создавать интерфейсы в Java и применять их в программах.