Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий
институт
Кафедра «Информатика»
кафедра

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

Java Core, наследование
тема

Вариант 11

Преподаватель — подпись, дата — А.С. Черниговский (инициалы, фамилия)

Студент КИ18-16б 031831229 (номер группы, зачетной книжки) — подпись, дата — В.А. Прекель (инициалы, фамилия)

Красноярск 2020

1 Цель работы

Ознакомиться с механизмом наследования в языке Java. Повторить основные языковые конструкции языка Java.

2 Общая постановка задачи

Для каждого варианта имеется набор из четырех сущностей. Необходимо выстроить иерархию наследования. В каждом классе (базовом и производных) должно быть минимум одно числовое и одно текстовое поле. При вводе числовых параметров обязательна проверка на число и на диапазон (даже если число может быть любое, проверку необходимо реализовать).

Для всех классов должны быть реализованы конструкторы (по умолчанию, с параметрами), методы equals(), hashCode(), toString();

Реализовать консольное Java-приложение, которое имеет простейшее пользовательское меню, состоящее как минимум из следующих пунктов:

- Добавить новый элемент. (Элементы должны добавляться в коллекцию элементов типа базового класса. Необходимо предусмотреть возможность добавления любого объекта производного класса в данную коллекцию.)
- Удалить элемент по индексу.
- Сравнение двух элементов на равенство (по индексам).
- Завершение работы приложения.

Вариант 11. ЭВМ, ПК, ноутбук, планшет.

3 Исходный код

Листинг 1 – Lab01/src/main/java/Program.java

```
import entities.Computer;
import entities.Laptop;
import entities.PersonalComputer;
import entities.Tablet;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.function.Function;


public class Program {
    private static final ArrayList<Computer> ComputerList = new ArrayList<>();
```

```
    public static void main(String[] args) {

        var reader = new BufferedReader(new InputStreamReader(System.in));

        boolean needBreak = false;
        while (!needBreak) {
            System.out.println("1 - Добавить элемент класса Computer");
            System.out.println("2 - Добавить элемент класса PersonalComputer");
            System.out.println("3 - Добавить элемент класса Laptop");
            System.out.println("4 - Добавить элемент класса Tablet");
            System.out.println("5 - Удалить элемент по индексу");
            System.out.println("6 - Сравнение двух элементов на равенство (по
индексам)");
            System.out.println("7 - Печать всех элементов");
            System.out.println("8 - Завершение работы приложения");

            var cmdNumber = ReadIntWithCheck("Введите номер команды: ", reader,
Program::CheckCmd);

            switch (cmdNumber) {
                case 1: {
                    var field1 = ReadIntWithCheck("Введите Field1 класса
Computer: ", reader, Program::CheckInt);
                    var name = ReadStringWithCheck("Введите Name класса
Computer: ", reader, Program::CheckString);

                    getComputerList().add(new Computer(field1, name));
                    break;
                }
                case 2: {
                    var field1 = ReadIntWithCheck("Введите Field1 класса
PersonalComputer: ", reader, Program::CheckInt);
                    var field2 = ReadIntWithCheck("Введите Field2 класса
PersonalComputer: ", reader, Program::CheckInt);
                    var name = ReadStringWithCheck("Введите Name класса
PersonalComputer: ", reader, Program::CheckString);
                    var name2 = ReadStringWithCheck("Введите Name2 класса
PersonalComputer: ", reader, Program::CheckString);

                    getComputerList().add(new PersonalComputer(field1, field2,
name, name2));
                    break;
                }
                case 3: {
                    var field1 = ReadIntWithCheck("Введите Field1 класса Laptop:
", reader, Program::CheckInt);
                    var field2 = ReadIntWithCheck("Введите Field2 класса Laptop:
", reader, Program::CheckInt);
                    var field3 = ReadIntWithCheck("Введите Field3 класса Laptop:
", reader, Program::CheckInt);
                    var name = ReadStringWithCheck("Введите Name класса Laptop:
", reader, Program::CheckString);
                    var name2 = ReadStringWithCheck("Введите Name2 класса
Laptop: ", reader, Program::CheckString);
                    var name3 = ReadStringWithCheck("Введите Name3 класса
Laptop: ", reader, Program::CheckString);

                    getComputerList().add(new Laptop(field1, field2, field3,
name, name2, name3));
                    break;
                }
                case 4: {
```

```
                    var field1 = ReadIntWithCheck("Введите Field1 класса Tablet:
", reader, Program::CheckInt);
                    var field2 = ReadIntWithCheck("Введите Field2 класса Tablet:
", reader, Program::CheckInt);
                    var field4 = ReadIntWithCheck("Введите Field4 класса Tablet:
", reader, Program::CheckInt);
                    var name = ReadStringWithCheck("Введите Name класса Tablet:
", reader, Program::CheckString);
                    var name2 = ReadStringWithCheck("Введите Name2 класса
Tablet: ", reader, Program::CheckString);
                    var name4 = ReadStringWithCheck("Введите Name4 класса
Tablet: ", reader, Program::CheckString);
                    var tablet = new Tablet(field1, field2, field4, name, name2,
name4);
                    getComputerList().add(tablet);
                    break;
                }
                case 5: {
                    var indexToDelete = ReadIntWithCheck("Введите индекс для
удаления: ", reader, Program::CheckIndex);
                    getComputerList().remove(indexToDelete);
                    break;
                }
                case 6: {
                    var index1ToCompare = ReadIntWithCheck("Введите первый
индекс для сравнения: ", reader, Program::CheckIndex);
                    var index2ToCompare = ReadIntWithCheck("Введите второй
индекс для сравнения: ", reader, Program::CheckIndex);
                    if
(getComputerList().get(index1ToCompare).equals(getComputerList().get(index2ToCom
pare))) {
                        System.out.println("Объекты равны");
                    } else {
                        System.out.println("Объекты не равны");
                    }
                    break;
                }
                case 7: {
                    for (var i : getComputerList()) {
                        System.out.println(i);
                    }
                    break;
                }
                case 8: {
                    needBreak = true;
                    break;
                }
            }

            System.out.println();
        }
    }

    private static int ReadIntWithCheck(String message, BufferedReader reader,
Function<Integer, Boolean> checker) {
        while (true) {
            try {
                System.out.print(message);
                var number = Integer.parseInt(reader.readLine());
                if (!checker.apply(number)) {
                    throw new Exception("Не в промежутке");
                }
                return number;
```

```
            } catch (Exception e) {
                System.err.println(e.getMessage());
            }
        }
    }

    private static boolean CheckCmd(int number) {
        return 1 <= number && number <= 8;
    }

    private static boolean CheckInt(int number) {
        return 0 <= number && number < 1000;
    }

    private static String ReadStringWithCheck(String message, BufferedReader
reader, Function<String, Boolean> checker) {
        while (true) {
            try {
                System.out.print(message);
                var string = reader.readLine();
                if (!checker.apply(string)) {
                    throw new Exception("Пустая строка");
                }
                return string;
            } catch (Exception e) {
                System.err.println(e.getMessage());
            }
        }
    }

    private static boolean CheckString(String string) {
        return string != null && !string.isBlank();
    }

    public static ArrayList<Computer> getComputerList() {
        return ComputerList;
    }

    private static boolean CheckIndex(int number) {
        return 0 <= number && number < getComputerList().size();
    }
}
```

Листинг 2 – Lab01/src/main/java/entities/Computer.java

```
package entities;

public class Computer {

    protected String Name;
    private int Field1;

    public Computer() {
        Field1 = 1;
        Name = "ComputerName";
    }

    public Computer(int field1, String name) {
        Field1 = field1;
        Name = name;
    }
```

```
    @Override
    public int hashCode() {
        int result = getField1();
        result = 31 * result + (getName() != null ? getName().hashCode() : 0);
        return result;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Computer)) return false;

        Computer computer = (Computer) o;

        if (getField1() != computer.getField1()) return false;
        return getName() != null ? getName().equals(computer.getName()) :
computer.getName() == null;
    }

    @Override
    public String toString() {
        return "Computer{" +
                "Name='" + Name + '\'' +
                ", Field1=" + Field1 +
                '}';
    }

    public int getField1() {
        return Field1;
    }

    public String getName() {
        return Name;
    }

    public void setField1(int field1) {
        Field1 = field1;
    }
}
```

Листинг 3 – Lab01/src/main/java/entities/PersonalComputer.java

```
package entities;

public class PersonalComputer extends Computer {
    private int Field2;

    private String Name2;

    public PersonalComputer() {
        super(1, "PersonalComputerName");
        Field2 = 2;
        Name2 = "PersonalComputerName2";
    }

    public PersonalComputer(int field1, int field2, String name, String name2) {
        super(field1, name);
        setField2(field2);
        setName2(name2);
    }
```

```
    @Override
    public int hashCode() {
        int result = super.hashCode();
        result = 31 * result + getField2();
        result = 31 * result + (getName2() != null ? getName2().hashCode() : 0);
        return result;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof PersonalComputer)) return false;
        if (!super.equals(o)) return false;

        PersonalComputer that = (PersonalComputer) o;

        if (getField2() != that.getField2()) return false;
        return getName2() != null ? getName2().equals(that.getName2()) :
that.getName2() == null;
    }

    @Override
    public String toString() {
        return "PersonalComputer{" +
                "Field2=" + Field2 +
                ", Name2='" + Name2 + '\'' +
                ", Name='" + Name + '\'' +
                '}';
    }

    public int getField2() {
        return Field2;
    }

    public void setField2(int field2) {
        Field2 = field2;
    }

    public String getName2() {
        return Name2;
    }

    public void setName2(String name2) {
        Name2 = name2;
    }

}
```

Листинг 4 – Lab01/src/main/java/entities/Laptop.java

```
package entities;

public class Laptop extends PersonalComputer {
    private int Field3;
    private String Name3;

    public Laptop() {
        super(1, 2, "LaptopName", "LaptopName2");
        Field3 = 3;
        Name3 = "LaptopName3";
```

```
    }

    public Laptop(int field1, int field2, int field3, String name, String name2,
String name3) {
        super(field1, field2, name, name2);
        setField3(field3);
        setName3(name3);
    }

    @Override
    public int hashCode() {
        int result = super.hashCode();
        result = 31 * result + getField3();
        result = 31 * result + (getName3() != null ? getName3().hashCode() : 0);
        return result;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Laptop)) return false;
        if (!super.equals(o)) return false;

        Laptop laptop = (Laptop) o;

        if (getField3() != laptop.getField3()) return false;
        return getName3() != null ? getName3().equals(laptop.getName3()) :
laptop.getName3() == null;
    }

    @Override
    public String toString() {
        return "Laptop{" +
                "Field3=" + Field3 +
                ", Name3='" + Name3 + '\'' +
                ", Name='" + Name + '\'' +
                '}';
    }

    public int getField3() {
        return Field3;
    }

    public void setField3(int field3) {
        Field3 = field3;
    }

    public String getName3() {
        return Name3;
    }

    public void setName3(String name3) {
        Name3 = name3;
    }

}
```

Листинг 5 – Lab01/src/main/java/entities/Tablet.java

```
package entities;
```

```
public class Tablet extends PersonalComputer {

    private int Field4;

    private String Name4;

    public Tablet() {
        super(1, 2, "TabletName", "TabletName2");
        Field4 = 4;
        Name4 = "TabletName4";
    }

    public Tablet(int field1, int field2, int field4, String name, String name2,
String name4) {
        super(field1, field2, name, name2);
        setField4(field4);
        setName4(name4);
    }

    @Override
    public int hashCode() {
        int result = super.hashCode();
        result = 31 * result + getField4();
        result = 31 * result + (getName4() != null ? getName4().hashCode() : 0);
        return result;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Tablet)) return false;
        if (!super.equals(o)) return false;

        Tablet tablet = (Tablet) o;

        if (getField4() != tablet.getField4()) return false;
        return getName4() != null ? getName4().equals(tablet.getName4()) :
tablet.getName4() == null;
    }

    @Override
    public String toString() {
        return "Tablet{" +
                "Field4=" + Field4 +
                ", Name4='" + Name4 + '\'' +
                ", Name='" + Name + '\'' +
                '}';
    }

    public int getField4() {
        return Field4;
    }

    public void setField4(int field4) {
        Field4 = field4;
    }

    public String getName4() {
        return Name4;
    }

    public void setName4(String name4) {
        Name4 = name4;
```

```
        }
    }
```