



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/07 Интеллектуальные системы анализа,
обработки и интерпретации больших данных**

О Т Ч Е Т

по лабораторной работе № 4

Вариант 12

Название: Внутренние классы и интерфейсы в Java

Дисциплина: Языки программирования для работы с большими данными

Студент

ИУ6-23М

(Группа)

(Подпись, дата)

П.А. Мартынюк

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

П.В. Степанов

(И.О. Фамилия)

Москва, 2022

Цель работы:

Получение навыков работы с внутренними классами и интерфейсами языка программирования Java.

Выполнение:

Задание 1:

1. Создать класс CD (mp3-диск) с внутренним классом, с помощью объектов которого можно хранить информацию о каталогах, подкаталогах и записях.
2. Создать класс Mobile с внутренним классом, с помощью объектов которого можно хранить информацию о моделях телефонов и их свойствах.

Листинг выполнения подзадачи 1 (файл Component.java)

```
package task1;

import java.util.Objects;

public class Component {
    int id;
    String name;
    boolean playable;
    int size;
    int duration;

    public Component() {
    }

    public Component(int id, String name, boolean playable, int size, int duration) {
        this.id = id;
        this.name = name;
        this.playable = playable;
        this.size = size;
        this.duration = duration;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Component component = (Component) o;
        return id == component.id && playable == component.playable && size == component.size
        && duration == component.duration && Objects.equals(name, component.name);
    }

    @Override
    public int hashCode() {
        return Objects.hash(id, name, playable, size, duration);
    }

    public boolean isFolder(){
        return !playable;
    }

    public void updateSize(int size){
        this.size += size;
    }

    public int getId() {
```

```

        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public boolean isPlayable() {
        return playable;
    }

    public void setPlayable(boolean playable) {
        this.playable = playable;
    }

    public int getDuration() {
        return duration;
    }

    public void setDuration(int duration) {
        this.duration = duration;
    }

    public int getSize() {
        return size;
    }

    public void setSize(int size) {
        this.size = size;
    }

    @Override
    public String toString() {
        if(playable){
            return "Component{" +
                "id=" + id +
                ", name=" + name +
                ", size=" + size +
                ", duration=" + duration +
                "} \n";
        } else {
            return "Component{" +
                "id=" + id +
                ", name=" + name +
                ", size=" + size +
                "} \n";
        }
    }
}

```

Листинг выполнения подзадачи 1 (файл CD.java)

```

package task1;

import java.util.HashMap;

public class CD {

    String name;

    public CD(String name) {
        this.name = name;
    }

    class Directory {

```

```

HashMap<Component, Component> child_parent;
Component root;

public Directory() {
    child_parent = new HashMap<>();
    root = new Component(0, "Folder 0", false, 3, 0);
}

public void addComponent(Component child, Component parent){
    if(parent.isFolder()) {
        child_parent.put(child, parent);
        parent.updateSize(child.getSize());
        while (child_parent.get(parent) != null){
            parent = child_parent.get(parent);
            parent.updateSize(child.getSize());
        }
        if(!parent.equals(root)) {
            root.updateSize(child.getSize());
        }
    } else {
        System.out.println("Invalid operation!");
    }
}

public void addComponent(Component child){
    this.addComponent(child, this.root);
}

@Override
public String toString() {
    return "directory{" +
        "child_parent=" + child_parent +
        '}';
}

@Override
public String toString() {
    return "CD{" + '\n' +
        "name=" + name + '\n' +
        '}';
}
}

```

Листинг выполнения подзадачи 1 (файл MainForCD.java)

```

package task1;

public class MainForCD {

    public static void main(String[] args) {

        CD disk = new CD("CD 2.0");
        CD.Directory dir = disk.new Directory();

        Component folder1 = new Component(1, "Folder 1", false, 3, 0 );
        Component folder2 = new Component(2, "Folder 2",false, 3, 0 );
        Component folder3 = new Component(3, "Folder 3",false, 3, 0 );

        dir.addComponent(folder1);
        dir.addComponent(folder2);
        dir.addComponent(folder3, folder2);

        Component file1 = new Component(4, "Queen - We are the Champions",true, 1500, 190 );
        Component file2 = new Component(5, "Imagine Dragons - Believer",true, 1800, 120 );
        Component file3 = new Component(6, "Nickelback - She keeps me up",true, 900, 50 );
        Component file4 = new Component(7, "Kipelov - Ya svoboden",true, 1700, 160 );
        Component file5 = new Component(8, "Alisa - Moyo pokolenie",true, 2100, 210 );

        dir.addComponent(file1);
        dir.addComponent(file2, folder1);
        dir.addComponent(file3, folder1);
        dir.addComponent(file4, folder2);
        dir.addComponent(file5, folder2);
    }
}

```

```

        System.out.println(dir);
    }
}

```

```

Run: MainForCD x
C:\Users\polin\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ
directory{child_parent={Component{id=7, name=Kipelov - Ya svoboden, size=1700, duration=160}
=Component{id=2, name=Folder 2, size=3806}
, Component{id=4, name=Queen - We are the Champions, size=1500, duration=190}
=Component{id=0, name=Folder 0, size=8012}
, Component{id=1, name=Folder 1, size=2703}
=Component{id=0, name=Folder 0, size=8012}
, Component{id=8, name=Alisa - Moyo pokolenie, size=2100, duration=210}
=Component{id=2, name=Folder 2, size=3806}
, Component{id=6, name=Nickelback - She keeps me up, size=900, duration=50}
=Component{id=1, name=Folder 1, size=2703}
, Component{id=5, name=Imagine Dragons - Believer, size=1800, duration=120}
=Component{id=1, name=Folder 1, size=2703}
, Component{id=2, name=Folder 2, size=3806}
=Component{id=0, name=Folder 0, size=8012}
, Component{id=3, name=Folder 3, size=3}
=Component{id=2, name=Folder 2, size=3806}
}}

Process finished with exit code 0

```

Рисунок 1 - Результат выполнения кода решения подзадачи 1

Листинг выполнения подзадачи 2 (файл Mobile.java)

```

package task1;

import java.util.ArrayList;

public class Mobile {
    String firm;
    ArrayList<Model> models;

    public Mobile() {
        models = new ArrayList<>();
    }

    public Mobile(String firm) {
        this.firm = firm;
        models = new ArrayList<>();
    }

    public void addModel(String name) {
        models.add(new Model(name));
    }

    @Override
    public String toString() {
        return "Mobile \n{" +
            "firm: '" + firm + '\'' +
            "\nmodels: \n" + models +
            '}' ;
    }

    class Model{
        String name;
    }
}

```

```

ArrayList<Attribute> attributes;

public Model() {
    attributes = new ArrayList<>();
}

public Model(String name) {
    this.name = name;
    attributes = new ArrayList<>();
}

public void addAttribute(String name, int amount){
    Attribute attribute = new Attribute(name, amount);
    attributes.add(attribute);
}

public void addAttribute(String name){
    Attribute attribute = new Attribute(name);
    attributes.add(attribute);
}

@Override
public String toString() {
    return "\n  Model{" +
        "name='" + name + '\'' +
        "\n    attributes=" + attributes + '\n' +
        "  }";
}

class Attribute{
    String name;
    int amount;

    public Attribute() {
    }

    public Attribute(String name) {
        this.name = name;
        this.amount = -1;
    }

    public Attribute(String name, int amount) {
        this.name = name;
        this.amount = amount;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAmount() {
        return amount;
    }

    public void setAmount(int amount) {
        this.amount = amount;
    }

    @Override
    public String toString() {
        if(amount != -1) {
            return "Attribute{" +
                "name='" + name + '\'' +
                ", amount=" + amount +
                '}';
        } else {
            return "Attribute{" +
                "name='" + name +
                '\'';
        }
    }
}

```

```
package task1;

public class MainForMobile {
    public static void main(String[] args) {

        Mobile mobile = new Mobile("Xiaomi");

        mobile.addModel("Mi 11");
        mobile.models.get(0).addAttribute("RAM", 8);
        mobile.models.get(0).addAttribute("ROM", 256);
        mobile.models.get(0).addAttribute("Accumulator", 4600);
        mobile.models.get(0).addAttribute("NFC");

        mobile.addModel("Mi 8");
        mobile.models.get(1).addAttribute("RAM", 6);
        mobile.models.get(1).addAttribute("ROM", 128);
        mobile.models.get(0).addAttribute("Accumulator", 3400);

        System.out.println(mobile);
    }
}
```

Рисунок 2 - Результат выполнения кода решения подзадачи 2

Реализовать абстрактные классы или интерфейсы, а также наследование и полиморфизм для следующих классов:

- ```
package task2;

public interface Abiturient {
 void passExam(String subject, int score);
}
```

## Листинг выполнения подзадачи 1 (файл Student.java)

```
package task2;

import java.util.HashMap;

public abstract class Student implements Abiturient{

 private String name;
 private String group;
 private HashMap<String, Integer> subjectScores;

 public Student() {
 subjectScores = new HashMap<>();
 }

 public Student(String name, String group) {
 this.name = name;
 this.group = group;
 subjectScores = new HashMap<>();
 }

 public void addSubject(String subject){
 subjectScores.put(subject, 0);
 }

 public void addPoints(String subject, int points){
 subjectScores.put(subject, subjectScores.get(subject) + points);
 }

 @Override
 public void passExam(String subject, int score) {
 int pointsPerSemester = subjectScores.get(subject);
 if (score < 18 || (pointsPerSemester + score) < 60){
 System.out.println(subject + " exam is not passed by " + name + "!");
 } else {
 subjectScores.put(subject, subjectScores.get(subject) + score);
 }
 }

 public String getName() {
 return name;
 }

 public void setName(String name) {
 this.name = name;
 }

 public String getGroup() {
 return group;
 }

 public void setGroup(String group) {
 this.group = group;
 }

 public HashMap<String, Integer> getSubjectScores() {
 return subjectScores;
 }

 public void setSubjectScores(HashMap<String, Integer> subjectScores) {
 this.subjectScores = subjectScores;
 }

 @Override
 public String toString() {
 return "name='" + name + '\'' +
 ", group='" + group + '\'' +
 ", subjectScores=" + subjectScores
 ;
 }
}
```



### Листинг выполнения подзадачи 1 (файл ExtramuralStudent.java)

```
package task2;

public class ExtramuralStudent extends Student{

 String typeOfExtramuralEducation;

 public ExtramuralStudent() {
 }

 public ExtramuralStudent(String name, String group, String typeOfExtramuralEducation) {
 super(name, group);
 this.typeOfExtramuralEducation = typeOfExtramuralEducation;
 }

 public String getTypeOfExtramuralEducation() {
 return typeOfExtramuralEducation;
 }

 public void setTypeOfExtramuralEducation(String typeOfExtramuralEducation) {
 this.typeOfExtramuralEducation = typeOfExtramuralEducation;
 }

 @Override
 public String toString() {
 return "ExtramuralStudent{" +
 "typeOfExtramuralEducation='" + typeOfExtramuralEducation +
 super.toString() + '\'' +
 '}';
 }
}
```

### Листинг выполнения подзадачи 1 (файл MainForStudent.java)

```
package task2;

public class MainForStudent {
 public static void main(String[] args) {

 ExtramuralStudent exSt1 = new ExtramuralStudent("Ivanov Maxim Andreevich", "ICS6-13",
"Moduled");
 ExtramuralStudent exSt2 = new ExtramuralStudent("Smirnova Ekaterina Ivanovna", "ICS5-
11", "Classic");

 exSt1.addSubject("Mathematics");
 exSt1.addPoints("Mathematics", 59);
 exSt1.addSubject("History");
 exSt1.addPoints("History", 55);

 exSt2.addSubject("Mathematics");
 exSt2.addPoints("Mathematics", 60);
 exSt2.addSubject("History");
 exSt2.addPoints("History", 53);

 System.out.println("Before exams:");
 System.out.println(exSt1);
 System.out.println(exSt2);

 System.out.println();
 System.out.println("After exams:");
 exSt1.passExam("Mathematics", 29);
 exSt1.passExam("History", 12);
 exSt2.passExam("Mathematics", 25);
 exSt2.passExam("History", 23);

 System.out.println(exSt1);
 System.out.println(exSt2);

 }
}
```

```
Run: MainForStudent
C:\Users\polin\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2021.3.2\lib\idea_rt.jar=59802:C:\Program Files
Before exams:
ExtramuralStudent{typeOfExtramuralEducation='Moduledname='Ivanov Maxim Andreevich', group='ICS6-13', subjectScores={Mathematics=59, History=55}}
ExtramuralStudent{typeOfExtramuralEducation='Classicname='Smirnova Ekaterina Ivanovna', group='ICS5-11', subjectScores={Mathematics=60, History=53}}
After exams:
History exam is not passed by Ivanov Maxim Andreevich!
ExtramuralStudent{typeOfExtramuralEducation='Moduledname='Ivanov Maxim Andreevich', group='ICS6-13', subjectScores={Mathematics=88, History=55}}
ExtramuralStudent{typeOfExtramuralEducation='Classicname='Smirnova Ekaterina Ivanovna', group='ICS5-11', subjectScores={Mathematics=85, History=70}}
Process finished with exit code 0
```

Рисунок 3 - Результат выполнения кода решения подзадачи 1

#### Листинг выполнения подзадачи 2 (файл Employee.java)

```
package task2;
```

```
public interface Employee {
 String introduce();
}
```

#### Листинг выполнения подзадачи 2 (файл Engineer.java)

```
package task2;
```

```
public class Engineer implements Employee{
 private String name;
 private String position;
 private String organization;
 private String department;

 public Engineer() {
 }

 public Engineer(String name, String position, String organization, String department) {
 this.name = name;
 this.position = position;
 this.organization = organization;
 this.department = department;
 }

 @Override
 public String introduce() {
 return "Hello, \nmy name is " + name + ",\nI am a " + position + " in " + organization
+ " at " + department + " department";
 }

 public String getName() {
 return name;
 }

 public void setName(String name) {
 this.name = name;
 }

 public String getPosition() {
 return position;
 }

 public void setPosition(String position) {
 this.position = position;
 }

 public String getOrganization() {
 return organization;
 }

 public void setOrganization(String organization) {
 this.organization = organization;
 }

 public String getDepartment() {
```

```

 return department;
 }

 public void setDepartment(String department) {
 this.department = department;
 }

 @Override
 public String toString() {
 return "Engineer{" +
 "name='" + name + '\'' +
 ", position='" + position + '\'' +
 ", organization='" + organization + '\'' +
 ", department='" + department + '\'' +
 '}';
 }
}

```

Листинг выполнения подзадачи 2 (файл Head.java)

```

package task2;

public class Head extends Engineer{

 public Head() {
 }

 public Head(String name, String position, String organization, String department) {
 super(name, position, organization, department);
 }

 @Override
 public String introduce() {
 return super.introduce() + ", I am head of this department";
 }
}

```

Листинг выполнения подзадачи 2 (файл MainForEngineer.java)

```

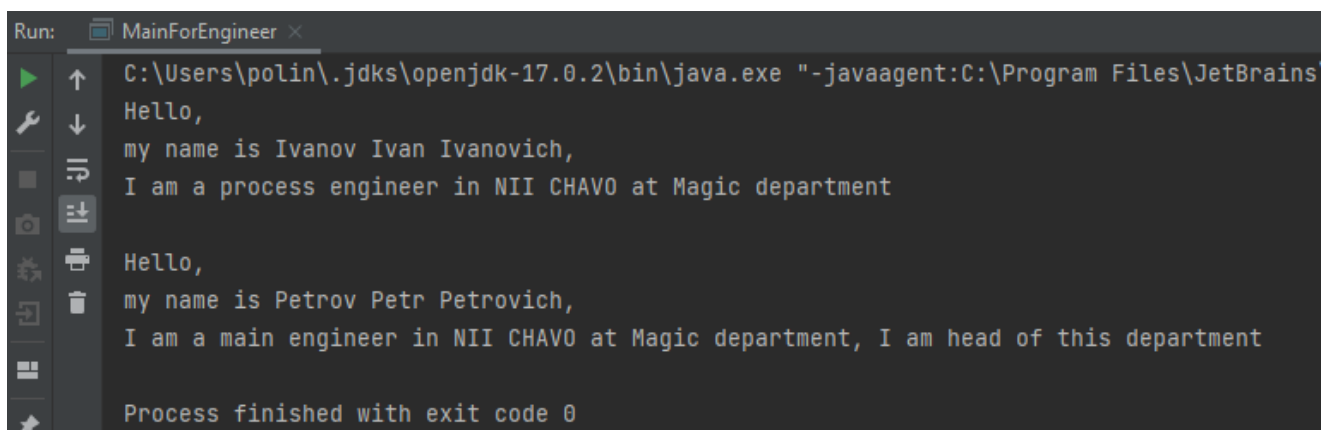
package task2;

public class MainForEngineer {
 public static void main(String[] args) {

 Engineer engineer = new Engineer("Ivanov Ivan Ivanovich", "process engineer", "NII CHAVO", "Magic");
 Head head = new Head("Petrov Petr Petrovich", "main engineer", "NII CHAVO", "Magic");

 System.out.println(engineer.introduce());
 System.out.println();
 System.out.println(head.introduce());
 }
}

```



```

Run: MainForEngineer x
C:\Users\polin\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains
Hello,
my name is Ivanov Ivan Ivanovich,
I am a process engineer in NII CHAVO at Magic department

Hello,
my name is Petrov Petr Petrovich,
I am a main engineer in NII CHAVO at Magic department, I am head of this department

Process finished with exit code 0

```

Рисунок 4 - Результат выполнения кода решения подзадачи 2

### **Ссылка на программное решение:**

Программное решение представлено в репозитории распределённой системы управления версиями Git:

<https://github.com/Owlfeather/JavaMagisterCourse/tree/main/Lab4/src>

### **Вывод:**

При выполнении лабораторной работы были получены навыки работы с внутренними классами и интерфейсами языка программирования Java.