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 18

Название: Внутренние классы и интерфейсы в Java

Дисциплина: Языки программирования для работы с большими данными

Студент ИУ6-23М (Группа) _____ (Подпись, дата) Д.Н. Хныкин (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 Computer (компьютер) с внутренним классом, с помощью объектов которого можно хранить информацию об операционной системе, процессоре и оперативной памяти.
2. Создать класс Park (парк) с внутренним классом, с помощью объектов которого можно хранить информацию об аттракционах, времени их работы и стоимости.

Листинг выполнения подзадачи 1

#Computer.java

```
package com.aranei.var1;

public class Computer {

    String name;
    TechSpecs specs;

    public Computer(String name, TechSpecs specs) {
        this.name = name;
        this.specs = specs;
    }
    public Computer(String os, String processor, int ram) {
        specs = new TechSpecs(os, processor, ram);
    }
    static class TechSpecs{
        String os;
        String processor;
        int ram;
        public TechSpecs(String os, String processor, int ram) {
            this.os = os;
            this.processor = processor;
            this.ram = ram;
        }

        public String getOs() {
            return os;
        }

        public void setOs(String os) {
            this.os = os;
        }

        public String getProcessor() {
            return processor;
        }

        public void setProcessor(String processor) {
            this.processor = processor;
        }

        public int getRam() {
            return ram;
        }

        public void setRam(int ram) {
            this.ram = ram;
```

```
        }

        @Override
        public String toString() {
            return "TechSpecs{" +
                    "os='" + os + '\'' +
                    ", processor='" + processor + '\'' +
                    ", ram=" + ram +
                    '}';
        }
    }
}
```

#Park.java

```
package com.aranei.var1;

public class Park {

   String name;
   Attraction attraction;


   public Park(String name, String work, int cost) {
      attraction = new Attraction(name, work, cost);
   }

   public Park(String name, Attraction attraction) {
      this.name = name;
      this.attraction = attraction;
   }


   static class Attraction{
      String name;
      String work;
      int cost;

      public Attraction(String name, String work, int cost) {
         this.name = name;
         this.work = work;
         this.cost = cost;
      }

      public String getName() {
         return name;
      }

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

      public String getWork() {
         return work;
      }

      public void setWork(String work) {
         this.work = work;
      }

      public int getCost() {
         return cost;
      }

      public void setCost(int cost) {
         this.cost = cost;
      }

      @Override
      public String toString() {
         return "Attraction{" +
               "name='" + name + '\'' +
               ", work='" + work + '\'' +
               ", cost=" + cost +
               '}';
      }
   }
} c
```

Задание 2:

3. interface Корабль <- class Грузовой Корабль <- class Танкер.
4. interface Мебель <- abstract class Шкаф <- class Книжный Шкаф.

Листинг выполнения подзадачи 2

#Ship.java

```
package com.aranei.var2.n8;

public interface Ship {
    void targetPort(String portName);
}
```

#CargoShip.java

```
package com.aranei.var2.n8;

public class CargoShip implements Ship{
    String portName;
    int carrying;

    @Override
    public void targetPort(String portName) {
        this.portName = portName;
    }
};

            for (int col = 0; col < matrix.cols; ++col) {
                Matrix sub = subMatrix(matrix, 1, col + 1);

                result += (Math.pow(-1, 1 + col + 1) *
                        matrix.data[0][col] * _determinant(sub));
            }
            return result;
        }
    }
}
```

#Tanker.java

```
package com.aranei.var2.n8;

public class Tanker extends CargoShip{

    String type;

    public Tanker(String type, int carrying, String portName) {
        this.type = type;
        this.carrying = carrying;
        this.portName = portName;
    }

    public String getType() {
        return type;
    }

    public void setType(String type) {
        this.type = type;
    }

    public void pushFuel(){
        this.type = "empty";
    }

}
```

#Furniture.java

```
package com.aranei.var2.n9;

public interface Furniture {

    void material(String type);
```

```
}
```

#Closet.java

```
package com.aranei.var2.n9;

public class Closet implements Furniture{
    String material;
    int height;
    int width;
    int depth;

    public Closet(int height, int width, int depth) {
        this.height = height;
        this.width = width;
        this.depth = depth;
    }

    public Closet() {
    }

    public Closet(String material, int height, int width, int depth) {
        this.material = material;
        this.height = height;
        this.width = width;
        this.depth = depth;
    }

    @Override
    public void material(String type) {
        this.material = type;
    }

    public int getHeight() {
        return height;
    }

    public void setHeight(int height) {
        this.height = height;
    }

    public int getWidth() {
        return width;
    }

    public void setWidth(int width) {
        this.width = width;
    }

    public int getDepth() {
        return depth;
    }

    public void setDepth(int depth) {
        this.depth = depth;
    }

    @Override
    public String toString() {
        return "Closet{" +
                "material='" + material + '\'' +
                ", height=" + height +
                ", width=" + width +
                ", depth=" + depth +
                '}';
    }
}
```

#Bookshelf.java

```
package com.aranei.var2.n9;

public class Bookshelf extends Closet{
    int shelves;

    public Bookshelf(int height, int width, int depth, int shelves) {
        super(height, width, depth);
        this.shelves = shelves;
```

```
        }

        public Bookshelf(int shelves) {
            this.shelves = shelves;
        }

        public Bookshelf(String material, int height, int width, int depth, int shelves) {
            super(material, height, width, depth);
            this.shelves = shelves;
        }


        public int getShelves() {
            return shelves;
        }

        public void setShelves(int shelves) {
            this.shelves = shelves;
        }

        public void addShelves(int num){
            this.shelves += num;
        }

        public void deleteShelves(int num){
            this.shelves -= num;
        }
    }
```

Ссылка на программное решение

Программное решение представлено в репозитории распределённой системы управления версиями Git:

https://github.com/Aranei99/labsJava/tree/main/LB4

Вывод

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