Задание 1

Реализовать указанный класс, включив в него вспомогательный внутренний класс или классы. Реализовать 2-3 метода (на выбор). Продемонстрировать использование реализованных классов.

Создать класс CD (mp3-диск) с внутренним классом, с помощью объектов которого можно хранить информацию о каталогах, подкаталогах и записях.

Код программы

Main.java

```
class Main {
  public static void main(String[] args) {
    Disk disk = new Disk();
    disk.createCatalog("first cat");
    disk.createCatalog("second cat");
    disk.createRecord("a.mp3");
    disk.showDiskFiles();

    System.out.println();
    disk.root.getCatalogs().elementAt(0).createCatalog("included in first cat");
    disk.root.getCatalogs().elementAt(0).createRecord("included in first cat record.mp3");
    disk.root.getCatalogs().elementAt(0).showCatalog();
  }
}
```

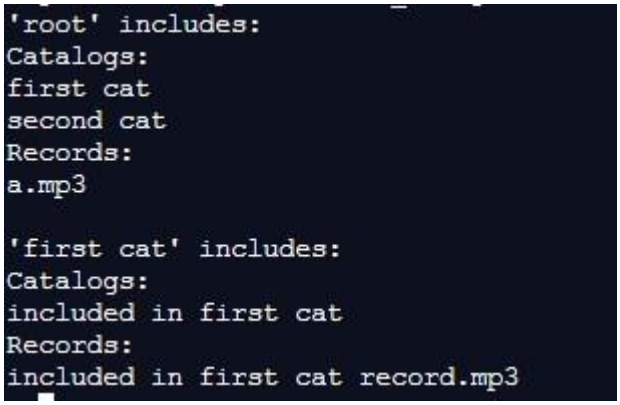
Disk.java

```
import java.util.*;
public class Disk{
  public class Catalog{
    public class Record{
      private String name;
      private Byte[] data;

      public Record(String name){
        this.name = name;
      }
      public String getName(){
        return name;
      }
      public Byte[] getData(){
        return data;
      }
    }

    public Catalog(String name){
      this.name = name;
      catalogs = new Vector<>();
      records = new Vector<>();
    }
```

```
 public Vector<Catalog> getCatalogs(){
  return catalogs;
 }
 public Vector<Record> getRecords(){
  return records;
 }
 public void deleteCatalog(String name){
  for(int i = 0; i<catalogs.size(); i++)
   if(catalogs.elementAt(i).getName().equals(name)){
    catalogs.remove(i);
    break;
   }
 }
 public void createCatalog(String name){
  for(int i = 0; i<catalogs.size(); i++)
   if(catalogs.elementAt(i).getName().equals(name))
    return;
  catalogs.addElement(new Catalog(name));
 }
 public void createRecord(String name){
  for(int i = 0; i<records.size(); i++)
   if(records.elementAt(i).getName().equals(name))
    return;
  records.addElement(new Record(name));
 }
 public String getName(){
  return name;
 }
 public void showCatalog(){
  System.out.println("\'" + name + "\' includes:");
  System.out.println("Catalogs:");
  for(Catalog c : catalogs)
   System.out.println(c.getName());
  System.out.println("Records:");
  for(Record r : records)
   System.out.println(r.getName());
 }

 private Vector<Catalog> catalogs;
 private Vector<Record> records;
 private String name;
}

public Disk(){
 root = new Catalog("root");
}

public void createCatalog(String name){
 root.createCatalog(name);
}
```

```
  public void createRecord(String name){
    root.createRecord(name);
  }
  public void showDiskFiles(){
    root.showCatalog();
  }

  public Catalog root;
}
```

Вывод программы



Задание 2

Реализовать агрегирование. При создании класса агрегируемый класс объявляется как атрибут (локальная переменная, параметр метода). Включить в каждый класс 2-3 метода на выбор. Продемонстрировать использование разработанных классов.
Создать класс Абзац, используя класс Слово

Код программы

Main.java

```
class Main {
  public static void main(String[] args) {
    Word w1 = new Word("i");
    Word w2 = new Word("did");
    Word w3 = new Word("this");
    Word w4 = new Word("laboratory");
    Word w5 = new Word("work");
    Abzac ab = new Abzac(w1,w2,w3,w4,w5);
    ab.print();
  }
}
```

Abzac.java

```
import java.util.*;

public class Abzac{
  public Abzac(){
    words = new Vector<>();
  }
  public Abzac(Word ..._words){
```

```
    words = new Vector<>();
    for(Word word : _words)
      words.addElement(word);
    words_count = words.size();
  }

  public void print(){
    if(words_count < 1 ){
      System.out.println("Nothing to print");
      return;
    }
    for(int i = 0; i<words.size(); i++){
      if(i == 0){
        System.out.print(words.elementAt(i).firstToUpper() + " ");
      } else {
        System.out.print(words.elementAt(i).toString() + " ");
      }
    }
    System.out.println();
  }
  public void addWord(String word){
    words.addElement(new Word(word));
    words_count = words.size();
  }
  public void addWord(Word word){
    words.addElement(word);
    words_count = words.size();
  }

  private Vector<Word> words;
  private int words_count;
}
```

Word.java

```
public class Word{
  private String word;
  private Byte[] meta;

  public Word(String _word){
    word = _word;
    //some meta
  }

  public String toUpper(){
    String w = "";
    for(int i = 0; i<word.length(); i++){
      w += Character.toUpperCase(word.charAt(i));
    }
    return w;
```

```
 }
 public String firstToUpper(){
  String w = word;
  char[] cw = w.toCharArray();
  cw[0] = Character.toUpperCase(word.charAt(0));
  w = String.valueOf(cw);
  return w;
 }
 public String toLower(){
  String w = "";
  for(int i = 0; i<word.length(); i++){
   w += Character.toLowerCase(word.charAt(i));
  }
  return w;
 }
 public void print(){
  System.out.println(word);
 }
 public String toString(){
  return word;
 }
}
```

```
I did this laboratory work
```

Задание 3

Построить модель программной системы с применением отношений (обобщения, агрегации, ассоциации, реализации) между классами. Задать атрибуты и методы классов. Реализовать (если необходимо) дополнительные классы. Продемонстрировать работу разработанной системы. Система Телефонная станция. Абонент оплачивает Счет за разговоры и Услуги, может попроситьАдминистраторасменитьномериотказатьсяотуслуг.Администраторизменяет номер, Услуги и временно отключает Абонента за неуплату.

Код программы

Main.java

```
class Main {
 public static void main(String[] args) {
  StationManager manager = new StationManager();
  User abonent = new User("+8800553535", false);
  User admin = new User("+8800880000", true);
  abonent.setMeta("This is good user");
  System.out.println("Abonent’s meta: " + abonent.getMeta());

  manager.addUser(abonent);
  manager.addUser(admin);
  manager.print(abonent);

  manager.pay(abonent, -10);
```

```
   manager.deactiveUser(admin, abonent);
   manager.print(abonent);
   manager.pay(abonent, 20);
   manager.print(abonent);

   manager.changeNumber(abonent, abonent, "+7700777777");
   manager.changeNumber(admin, abonent, "+7700777777");
   manager.print(abonent);

   manager.addService(admin, abonent, "Online TV", 5.3);
   manager.print(abonent);

   manager.deleteService(admin, abonent, "Online TV");
   manager.deleteService(admin, abonent, "Online TV");
   manager.print(abonent);
  }
}
```

StationManager.java

```
import java.util.*;

public class StationManager{

  private Vector<User> users;

  public StationManager(){
    users = new Vector<>();
  }

  public void changeNumber(User user, User target, String newNumber){
    if(checkPerm(user))
      target.changeNumber(newNumber);
  }
  public void addService(User user, User target, String name, double price){
    if(checkPerm(user))
      target.addService(name, price);
  }
  public void deleteService(User user, User target, String name){
    if(checkPerm(user))
      target.deleteService(name);
  }
  public void deactiveUser(User user, User target){
    if(checkPerm(user))
      target.setActive(false);
  }
  private boolean checkPerm(User user){
    if(user.isAdmin()){
      return true;
    } else {
      System.out.println("You dont have permissions. Please contact your administrator\n");
```

```
      return false;
    }
  }

  public void printAmount(User user){
    System.out.println("\'" + user.getNumber() + "\' : " + user.getAmount());
  }

  public void pay(User user, double value){
    user.addVallet(value);
  }
  public void addUser(User user){
    users.addElement(user);
  }
  public void print(User user){
    for(User u : users){
      if(u.equals(user)){
        u.print();
      }
    }
  }
}
```

User.java

```
  import java.util.*;

  public class User<T>{
    public abstract class Service{
      private int type;
      private double price;
      private String name;

      public Service(){
      }

      public Service(String _name, int _type, double _price){
        name = _name;
        type = _type;
        price = _price;
      }
      public double getPrice(){
        return price;
      }

      public int getType(){
        return type;
      }

      public String getName(){
        return name;
```

```
  }
}
public class ExtraService extends Service{
  public ExtraService(String _name, double _price){
    super.Service(_name, 2, _price);
  }
}
public class Calls extends Service{
  public Calls(String _name){
    super.Service(_name, 2, 10.0);
  }
}
private T someMeta;
private boolean isAdmin;
private boolean isActive;
private double vallet;
private String number;
private Vector<Service> services;

public User(String _number, boolean _isAdmin){
  number = _number;
  isAdmin = _isAdmin;
  vallet = 0;
  isActive = true;
  services = new Vector<>();
  if(!isAdmin)
    services.add(new Calls("Calls"));
}
public void setMeta(T value){
  someMeta = value;
}
public T getMeta(){
  return someMeta;
}
public void changeNumber(String newNumber){
  number = newNumber;
}
public boolean isAdmin(){
  return isAdmin;
}
public boolean isActive(){
  return isActive;
}
public void setActive(boolean active){
  isActive = active;
}
public String getNumber(){
  return number;
}
public void addVallet(double value){
```

```
    vallet += value;
    if(!isActive && vallet > 0)
      isActive = true;
  }
  public void addService(String name, double price){
    services.add(new ExtraService(name, price));
  }
  public double getAmount(){
    double total = 0;
    for(Service s : services){
      total += s.getPrice();
    }
    return total;
  }

  public void print(){
    System.out.println(
      (isAdmin ? "Administrator. " : "Abonent. ") +
      "Number: " + number);
      if(!isAdmin){
        String serv=  "Services: \n";
        for(Service s : services)
            serv+= s.getName() + "("+s.getPrice()+")\n";
        System.out.print(serv);
        System.out.println("Vallet: " + vallet);
      }
      System.out.println(isActive ? "Active\n" : "Deactivated\n");
  }
  public void deleteService(String name){
    for(int i = 0; i<services.size(); i++){
      if(services.elementAt(i).getName().equals(name)){
        services.remove(i);
        return;
      }
    }
    System.out.println("Service not found\n");
  }

}
```

Вывод программы

Информация о пользователе

```
Abonent. Number: +8800553535
Services:
Calls(10.0)
Vallet: 0.0
Active
```

Измененный статус пользователя(деактивирован)

```
Abonent. Number: +8800553535
Services:
Calls(10.0)
Vallet: -10.0
Deactivated
```

Пополнение счета

```
Abonent. Number: +8800553535
Services:
Calls(10.0)
Vallet: 10.0
Active
```

Попытка изменить номер от имени абонента

```
You dont have permissions. Please contact your administrator
```

Изменение номера администратором

```
Abonent. Number: +7700777777
Services:
Calls(10.0)
Vallet: 10.0
Active
```

Добавление услуги

```
Abonent. Number: +7700777777
Services:
Calls(10.0)
Online TV(5.3)
Vallet: 10.0
Active
```

Удаление услуги

```
Abonent. Number: +7700777777
Services:
Calls(10.0)
Vallet: 10.0
Active
```

Попытка удаления отсутствующей услуги

```
Service not found
```

Вывод: приобрел практические навыки в области ООП.