Министерство образования Республики Беларусь

Учреждение образования «Брестский государственный технический университет»

Кафедра ИИТ

Отчёт по лабораторной работе № 3

По дисциплине «Современные платформы программирования»

Выполнил:
студент 3-го курса
группы ПО-9(2)
Николайчик Н.С.
Проверил:
Крощенко А. А.

Брест, 2024

Вариант 3

Цель работы: научиться создавать и использовать классы в программах на языке программирования Java.

Задание 1

Реализовать простой класс. Требования к выполнению
• Реализовать пользовательский класс по варианту.
• Создать другой класс с методом main, в котором будут находится примеры использования пользовательского класса. Для каждого класса
• Создать поля классов
• Создать методы классов
• Добавьте необходимые get и set методы (по необходимости)
• Укажите соответствующие модификаторы видимости
• Добавьте конструкторы
• Переопределить методы toString() и equals()

3) Прямоугольный треугольник, заданный длинами сторон – Предусмотреть возможность определения площади и периметра, а так же логический метод, определяющий существует или такой треугольник. Конструктор должен позволить создавать объекты с начальной инициализацией. Реализовать метод equals, выполняющий сравнение объектов данного типа.

Код класса:

```
public class Triangle {
   protected double catet_a,catet_b;

   public Triangle() {
      this.catet_a=5;
      this.catet_b=5;
   }
   public Triangle(Triangle tri) {
      this.catet_a=tri.catet_a;
      this.catet_b=tri.catet_b;
   }
   public Triangle(double a,double b) {
      this.catet_a=a;
      this.catet_b=b;
   }

   public double getCatet_a() {
      return catet_a;
   }

   public double getCatet_b() {
```

```
        return catet_b;
    }

    public void setCatet_a(double catet_a) {
        this.catet_a = catet_a;
    }

    public void setCatet_b(double catet_b) {
        this.catet_b = catet_b;
    }

    public double perimeter(){
        return this.catet_a + this.catet_b +
                Math.sqrt(this.catet_a*this.catet_a + this.catet_b*this.catet_b);
    }
    public double square(){
        return this.catet_b*this.catet_a/2;
    }
    public void output(){
        System.out.println(this.perimeter());
        System.out.println(this.square());
    }
    public boolean equals(Triangle a){
        if(this.catet_a==a.catet_a && this.catet_b==a.catet_b)
            return  true;
        return false;
    }
    public String toString(){
        return Double.toString(this.catet_a) + ";" + Double.toString(this.catet_b);
    }
}
```

Пример работы:

```
Введите длину катетов
4 5
Конструктор по-умолчанию
PERIMETER17.071067811865476
S12.5
Конструктор с параметрами
PERIMETER15.403124237432849
S10.0
Конструктор копирования
PERIMETER15.403124237432849
S10.0
```

Задание 2

Разработать автоматизированную систему на основе некоторой структуры дан- ных, манипулирующей объектами пользовательского класса. Реализовать требуемые функции обработки данных Требования к выполнению

• Задание посвящено написанию классов, решающих определенную задачу автоматизации;

• Данные для программы загружаются из файла (формат произволен). Файл создать и написать вручную.

Автоматизированная система в автобусном парке Составить программу, которая содержит информацию о наличие автобусов в автобусном парке. Сведения о каждом автобусе содержат (Bus) содержат:

• Фамилия и инициалы водителя;
• Номер автобуса;
• Номер маршрута;
• Марка;
• Год начала эксплуатации;
• Пробег;
• Местонахождение в настоящий момент времени (парк/маршрут).

Программа должна обеспечивать:

• Формирование данных обо всех автобусах в виде списка;
• Формирование списка автобусов выехавших из парка;
• Формирование списка автобусов оставшихся в парке;
• Список автобусов для заданного номера маршрута;
• Список автобусов, которые эксплуатируются больше 10 лет;
• Список автобусов, пробег у которых больше 100000 км.
• Вывод сведений об автобусах, находящихся на маршруте и об автобусах, оставшихся в парке.

Код:

```
import java.util.*;

public class Autopark {
   ArrayList<Bus> busList;

   public Autopark(){
      this.busList = new ArrayList<Bus>();
```

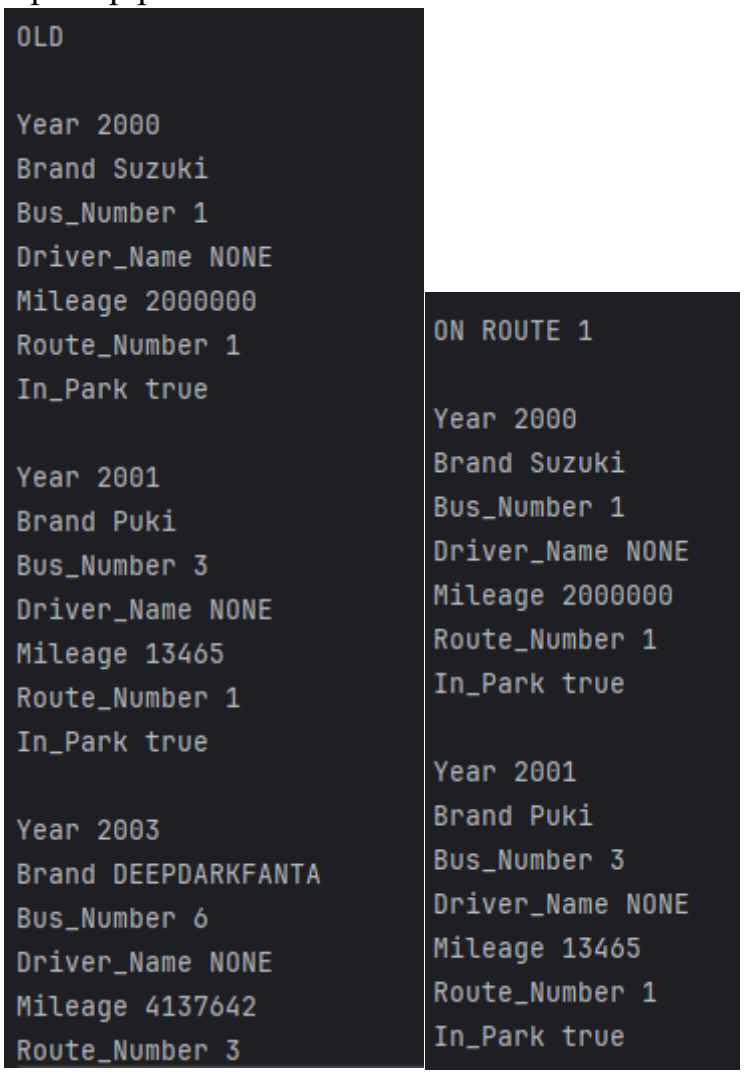
```
}

public ArrayList<Bus> allBus(){
   ArrayList<Bus> temp = new ArrayList<Bus>();
   for (int i = 0; i < this.busList.size(); i++) {
      temp.add(new Bus(busList.get(i)));
   }
   return temp;
}

public ArrayList<Bus> allBusNotInPark(){
   ArrayList<Bus> temp = new ArrayList<Bus>();
   for (int i = 0; i < this.busList.size(); i++) {
      if(!busList.get(i).isIn_Park()){
         temp.add(new Bus(busList.get(i)));
      }
   }
   return temp;
}

public ArrayList<Bus> allBusInPark(){
   ArrayList<Bus> temp = new ArrayList<Bus>();
   for (int i = 0; i < this.busList.size(); i++) {
      if(busList.get(i).isIn_Park()){
         temp.add(new Bus(busList.get(i)));
      }
      else continue;
   }
   return temp;
}

public ArrayList<Bus> allBusOnRoute(int Route){
   ArrayList<Bus> temp = new ArrayList<Bus>();
   for (int i = 0; i < this.busList.size(); i++) {
      if(busList.get(i).getRoute_Number() == Route){
         temp.add(new Bus(busList.get(i)));
      }
   }
   return temp;
}

public ArrayList<Bus> allBusOld(){
   ArrayList<Bus> temp = new ArrayList<Bus>();
   for (int i = 0; i < this.busList.size(); i++) {
      if( new Date().getYear() + 1900 - busList.get(i).getYear()  > 10){
         temp.add(new Bus(busList.get(i)));
      }
   }
   return temp;
```

```
    }

    public ArrayList<Bus> allBusManyKm(){
        ArrayList<Bus> temp = new ArrayList<Bus>();
        for (int i = 0; i < this.busList.size(); i++) {
            if(busList.get(i).getMileage()  >= 100000){
                temp.add(new Bus(busList.get(i)));
            }
        }
        return temp;
    }

    public void out_NotInPark(){
        ArrayList<Bus> temp = allBusNotInPark();
        for (int i = 0; i < temp.size(); i++) {
            temp.get(i).output();
        }
    }

    public void out_InPark(){
        ArrayList<Bus> temp = this.allBusInPark();
        for (int i = 0; i < temp.size(); i++) {
            temp.get(i).output();
        }
    }

    static public void out(ArrayList<Bus> temp){
        for (int i = 0; i < temp.size(); i++) {
            temp.get(i).output();
        }
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
    }
    void AddBus(int Bus_Number, int Route_Number , String Brand, int Year, long Mileage){
        this.busList.add(new Bus(Bus_Number,  Route_Number ,  Brand,  Year,  Mileage));
    }
}
```

```
public class Bus {

   String Driver_Name;
   int Bus_Number;
   int Route_Number;
   String Brand;
   int Year;
   long Mileage;
   boolean In_Park;

   public Bus(int Bus_Number, int Route_Number , String Brand, int Year, long Mileage){

      this.Driver_Name = "NONE";
      this.Bus_Number = Bus_Number;
      this.Route_Number = Route_Number;
      this.Brand = Brand;
      this.Year = Year;
      this.Mileage = Mileage;
      this.In_Park = true;
   }

   public Bus(Bus temp){

      this.Driver_Name = temp.Driver_Name;
      this.Bus_Number = temp.Bus_Number;
      this.Route_Number = temp.Route_Number;
      this.Brand = temp.Brand;
      this.Year = temp.Year;
      this.Mileage = temp.Mileage;
      this.In_Park = temp.In_Park;
   }

   public void setRoute_Number(int route_Number) {
      Route_Number = route_Number;
   }

   public void setDriver_Name(String driver_Name) {
      Driver_Name = driver_Name;
   }

   public void setIn_Park() {
      In_Park = !In_Park;
   }

   public long getMileage() {
      return Mileage;
   }

   public int getRoute_Number() {
```

```
        return Route_Number;
    }

    public int getYear() {
        return Year;
    }

    public boolean isIn_Park() {
        return In_Park;
    }

    public int getBus_Number() {
        return Bus_Number;
    }

    public String getBrand() {
        return Brand;
    }

    public String getDriver_Name() {
        return Driver_Name;
    }
    public void output(){
        System.out.println("Year " + this.Year);
        System.out.println("Brand " + this.Brand);
        System.out.println("Bus_Number "+this.Bus_Number);
        System.out.println("Driver_Name "+this.Driver_Name);
        System.out.println("Mileage "+this.Mileage);
        System.out.println("Route_Number "+this.Route_Number);
        System.out.println("In_Park "+ this.In_Park);
        System.out.println();

    }
}
```

Пример работы



Вывод: я научился использовать Java классы.