

Класс CarParking

```
1 package package3;
2
3 public class CarParking {
4     public int maxCarsNumber = 0;
5     private CarPlace[] carPlaces;
6     private int counter = 0;
7     public CarParking(int maxCarsNumber)
8     {
9
10        this.maxCarsNumber = maxCarsNumber;
11
12        carPlaces = new CarPlace[this.maxCarsNumber];
13
14        for(int i = 0; i < maxCarsNumber; i++){
15            carPlaces[i] = new CarPlace(i+1);
16        }
17    }
18
19    public void addCar(Car car)
20    {
21        if(counter < maxCarsNumber){
22            carPlaces[counter].vehicle = car;
23            counter++;
24        }
25        else
26            System.out.println("Все места заняты!");
27    }
28
29
30    public void sortByEngine()
31    {
32        //Сортировка "пузырьком"
33        CarPlace tempCarPlace;
34        for(int i = 0; i < maxCarsNumber; i++)
35        {
36            for(int j=maxCarsNumber-1;j>i;j--)
37            {
38                if(carPlaces[j-1].vehicle.getEngine() > carPlaces[j].vehicle.getEngine())
39                {
```

```

39         {
40             tempCarPlace = carPlaces[j-1];
41             carPlaces[j-1] = carPlaces[j];
42             carPlaces[j] = tempCarPlace;
43         }
44     }
45 }
46
47
48 public void sortById()
49 {
50     //Сортировка выбором
51     int minValueIndex ;
52     CarPlace tempPlace;
53     for(int i = 0; i < maxCarsNumber; i++)
54     {
55         minValueIndex = i; // Выбираем за минимальное значение первый не сортированный элемент
56         for(int j = i+1; j < maxCarsNumber; j++ )
57         {
58             if(carPlaces[j].id < carPlaces[minValueIndex].id) // Если элемент меньше принятого за наименьшее
59             {
60                 minValueIndex = j; // то меняем индекс наименьшего значения
61             }
62         }
63         tempPlace = carPlaces[i]; // Устанавливаем найденное значение под индексом i
64         carPlaces[i] = carPlaces[minValueIndex];
65         carPlaces[minValueIndex] = tempPlace;
66     }
67 }
68
69 @Override
70 public String toString()
71 {
72     String result = "";
73     for(int i = 0; i < maxCarsNumber; i++)
74     {
75         if(null != carPlaces[i].vehicle)
76             result += carPlaces[i].vehicle.toString() + "; Номер места: " + carPlaces[i].id + "\n";
77         else

```

```
77         else
78             result+="Пыто!";
79     }
80     return result;
81 }
82 }
83
```

Класс BusParking

```
1 package package3;
2
3 import java.util.ArrayList;
4
5
6 public class BusParking {
7     public int maxBusesNumber = 0;
8     private ArrayList<BusPlace> busPlaces;
9     private int counter = 0;
10
11     public BusParking(int maxBusesNumber)
12     {
13
14         this.maxBusesNumber = maxBusesNumber;
15
16         busPlaces = new ArrayList<BusPlace>();
17
18     }
19
20
21     public void addBus(Bus bus)
22     {
23         if(busPlaces.size() < maxBusesNumber)
24         {
25             BusPlace tempBusPlace = new BusPlace(counter+1);
26             tempBusPlace.vehicle=bus;
27             busPlaces.add(tempBusPlace);
28             counter++;
29
30         }
31     }
32
33     public void sortByEngine()
34     {
35         //Сортировка пузырьком
36         BusPlace tempBusPlace;
37         for(int i = 0; i < maxBusesNumber; i++) //выполняем n проходов по массиву, где n-размер массива
38         {
39             for(int j = maxBusesNumber-1 ; j > i ; j--) //проходим по массиву и сравниваем соседние элементы
```

```

39     for(int j = maxBusesNumber-1 ; j > i ; j--) //проходим по массиву и сравниваем соседние элемент
40     {
41         if(busPlaces.get(j-1).vehicle.getEngine() > busPlaces.get(j).vehicle.getEngine())
42         {
43             //Если элемент j больше чем j+1, то меняем их местами
44             tempBusPlace = busPlaces.get(j-1);
45             busPlaces.set(j-1, busPlaces.get(j));
46             busPlaces.set(j, tempBusPlace);
47         }
48     }
49 }
50
51
52 public void sortById()
53 {
54     //Сортировка выбором
55     int minValueIndex ;
56     BusPlace tempPlace;
57     for(int i = 0; i < maxBusesNumber; i++)
58     {
59         minValueIndex = i; // Выбираем за минимальное значение первый не сортированный элемент
60         for(int j = i+1; j < maxBusesNumber; j++ )
61         {
62             // Если элемент меньше принятого за наименьшее значение,
63             if(busPlaces.get(j).id < busPlaces.get(minValueIndex).id)
64             {
65                 minValueIndex = j; // то меняем индекс наименьшего значения
66             }
67         }
68         tempPlace = busPlaces.get(i); // Устанавливаем найденное значение под индексом i
69         busPlaces.set(i, busPlaces.get(minValueIndex));
70         busPlaces.set(minValueIndex, tempPlace);
71     }
72 }
73
74 @Override
75 public String toString()
76 {
77     String result = "";

```

```
77     String result = "";
78     for(int i = 0; i < busPlaces.size(); i++)
79     {
80         if(busPlaces.get(i).vehicle != null )
81             result+=busPlaces.get(i).vehicle.toString()+ "; Homep мeтpa: " + busPlaces.get(i).id +"\n";
82         else
83             result+="Нyчтo!";
84     }
85     return result;
86 }
87 }
```



Класс Vehicle

```

1 package package3;
2 //реализация интерфейса Comparable, используется для сравния двух объектов.
3 //Будет использоваться в пакете с коллекциями.
4 public class Vehicle implements Comparable{
5     private String name;
6     private String color;
7     private int engine;
8     private String gosNumber;
9
10    public Vehicle(String name,String color,String gosNumber,int engine)
11    {
12        this.name=name;
13        this.color=color;
14        this.engine=engine;
15        this.gosNumber=gosNumber;
16    }
17
18    public String toString()
19    {
20        return "Траснпортное средство "+getName()+" цвет:"+getColor()+" , государственный номер: "
21            +getGosNumber()+" , двигатель "+getEngine()+" лс";
22    }
23
24    //Инкапсуляция
25    public String getName() {
26        return name;
27    }
28    public void setName(String name) {
29        this.name = name;
30    }
31    public String getColor() {
32        return color;
33    }
34    public void setColor(String color) {
35        this.color = color;
36    }
37    public int getEngine() {
38        return engine;
39    }
40    public void setEngine(int engine) {

```

```
39 public void setEngine(int engine) {
40     this.engine = engine;
41 }
42 public String getGosNumber() {
43     return gosNumber;
44 }
45 public void setGosNumber(String gosNumber) {
46     this.gosNumber = gosNumber;
47 }
48
49 @Override
50 public int compareTo(Object o) {
51     // Сравнение двух объектов. Если они равны, то метод возвращает 0
52     // Если данный объект "больше" чем сравниваемый, то метод вернет положительный результат
53     // Наоборот-отрицательный
54     if(this.gosNumber.equals(((Vehicle)o).getGosNumber()))
55     {
56         return 0;
57     }
58     return this.gosNumber.compareTo(((Vehicle)o).getGosNumber());
59 }
60 @Override
61 public int hashCode()
62 {
63     return this.hashCode();
64 }
65 @Override
66 public boolean equals(Object o)
67 {
68     Vehicle v=(Vehicle) o;    //Приведение типов
69     if(this.name.equals(v.getName()) && this.gosNumber.equals(v.getGosNumber()))
70     {
71         return true;
72     }
73     return false;
74 }
75 }
76 }
```

Класс Main

```
1 package package3;
2
3 public class main {
4     public static void main(String[] args) {
5         // CarParking основанный на массиве
6         CarParking cp = new CarParking(3);
7
8         Car car=new Car("Chevrolet", "Белый", "95", 245, "sedan");
9         Car car2=new Car("Chevrolet2", "Белый", "95", 210, "sedan");
10        Car car3=new Car("Chevrolet3", "Белый", "95", 231, "sedan");
11
12        cp.addCar(car);
13        cp.addCar(car2);
14        cp.addCar(car3);
15
16        System.out.println(cp);
17
18        cp.sortByEngine();
19
20        System.out.println("Отсортированный по мощности двигателя массив :");
21        System.out.println(cp);
22
23        cp.sortById();
24
25        System.out.println("Отсортированный по номеру парковочного места массив :");
26        System.out.println(cp);
27
28        // BusParking основан на ArrayList
29        System.out.println("//-----//");
30
31        Bus ikarus=new Bus("Ikarus", "Зеленый", "v543gg", 192, 165F, 37);
32        Bus gazel=new Bus("Gazel", "Красный", "v783fg", 152, 4.8F, 6);
33        Bus max = new Bus("Maz", "Белый", "v346rt", 157, 143F, 30);
34
35
36        BusParking bp = new BusParking(3);
37        bp.addBus(max);
38        bp.addBus(gazel);
39        bp.addBus(ikarus);
```

```
39     bp.addBus(ikarus);
40
41
42     System.out.println(bp);
43
44     bp.sortByEngine();
45
46     System.out.println("Отсортированный по мощности двигателя массив :");
47     System.out.println(bp);
48
49     bp.sortById();
50
51     System.out.println("Отсортированный по номеру парковочного места массив :");
52     System.out.println(bp);
53
54 }
55 }
```