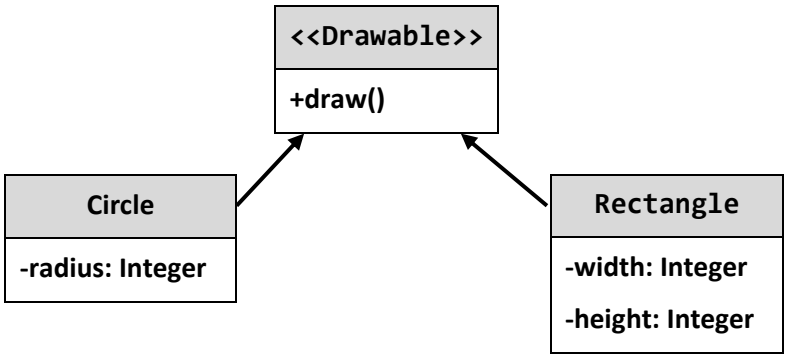
Lab: Interfaces and Abstraction

Problems for exercises and homework for the "Java OOP Advanced" course @ SoftUni.

You can check your solutions here: https://judge.softuni.bg/Contests/498/Interfaces-and-Abstraction-Lab .

1. Shapes Drawing

Build hierarchy of **interfaces** and **classes**:



You should be able to use the class like this:

Main.java

```
public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    Queue<Integer> queue = new ArrayDeque<>();
    for (int i = 0; i < 5; i++) {
        queue.add(Integer.parseInt(scanner.nextLine()));
    }

    Drawable circle = new Circle(queue.poll(), queue.poll(), queue.poll());
    Drawable rect = new Rectangle(queue.poll(), queue.poll());

    circle.draw();
    rect.draw();
}
```

Examples

Input	Output

4
6
6
5
4
	`      *******`
	`   ***        ***`
	`  **            **`
	`**                **`
	`*                  *`
	`**                **`
	`  **            **`
	`   ***        ***`
	`      *******`
	`* * * * *`
	`*       *`
	`*       *`
	`* * * * *`

Solution

For **circle** drawing you can use this algorithm:

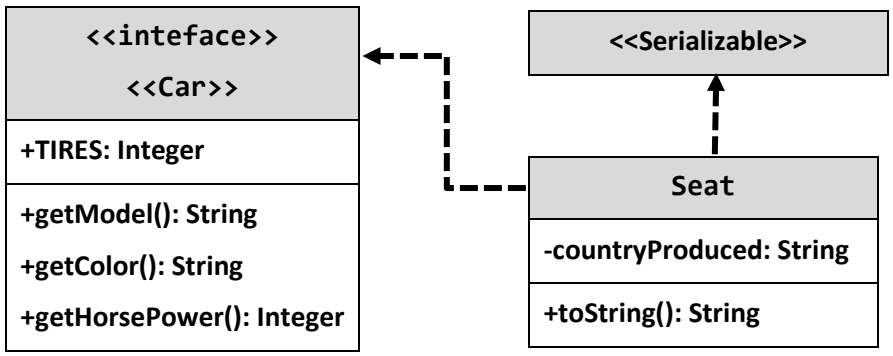
```
public void draw() {
    double r_in = this.radius - 0.4;
    double r_out = this.radius + 0.4;

    for(double y = this.radius; y >= -this.radius; --y)
    {
        for(double x = -this.radius; x < r_out; x += 0.5)
        {
            double value = x * x + y * y;
            if(value >= r_in * r_in && value <= r_out * r_out) {
                System.out.print("*");
            } else {
                System.out.print(" ");
            }
        }
        System.out.println();
    }
}
```

For **rectangle** drawing algorithm will be:

```
public void draw() {
    for (int i = 0; i < height; i++) {
        System.out.print("*");
        for (int k = 1; k < width - 1; k++) {
            System.out.print(" ");
            if (i == 0 || i == (height - 1)) {
                System.out.print("*");

            } else {
                System.out.print(" ");

            }
        }
        System.out.print(" ");
        System.out.print("*");
        System.out.print("\n");
    }
}
```

2. Car Shop

Build hierarchy from **classes** and **interfaces** for this UML diagram



Your hierarchy have to be used with this code

Main.java

```
public static void main(String[] args) {
    Car seat = new Seat("Leon", "gray", 110, "Spain");

    System.out.println(String.format(
            "%s is %s color and have %s horse power",
            seat.getModel(),
            seat.getColor(),
            seat.getHorsePower()));
    System.out.println(seat.toString());
}
```

Examples

Input	Output
	Leon is gray and have 110 horse power This is Leon produced in Spain and have 4 tires

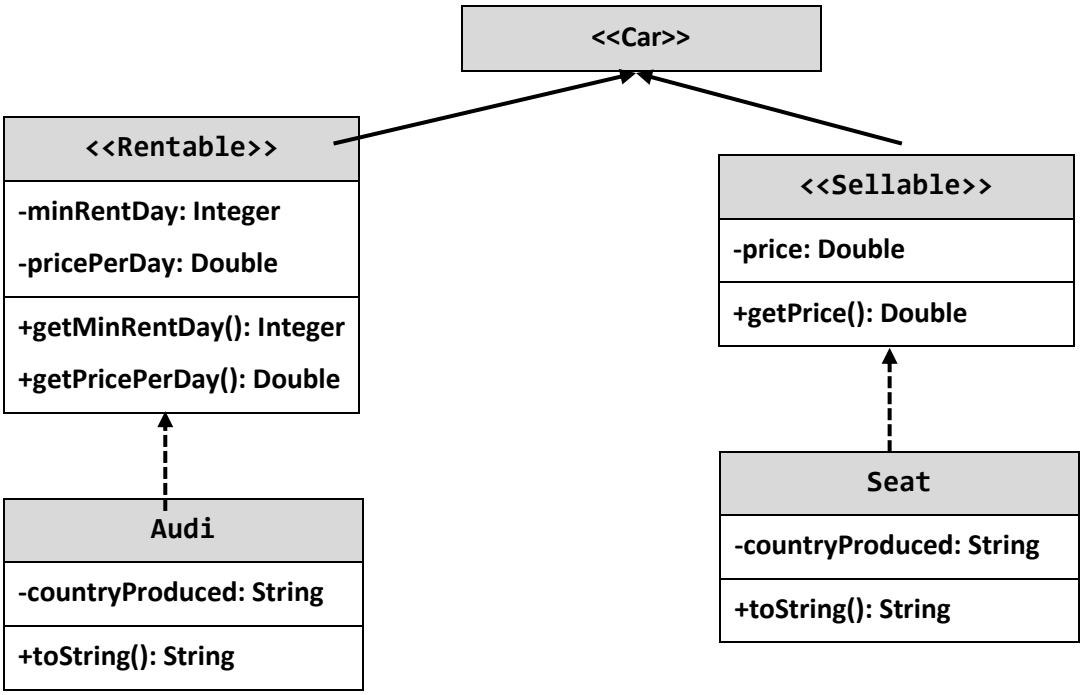
Solution

```
public interface Car {
    int TIRES = 4;

    String getModel();

    String getColor();

    int getHorsePower();
}
```

Note: consider using the wrapper classes in the **Seat** constructor.

3. Car Shop Extend

Extend previous problem:



Your hierarchy have to be used with this code

Main.java

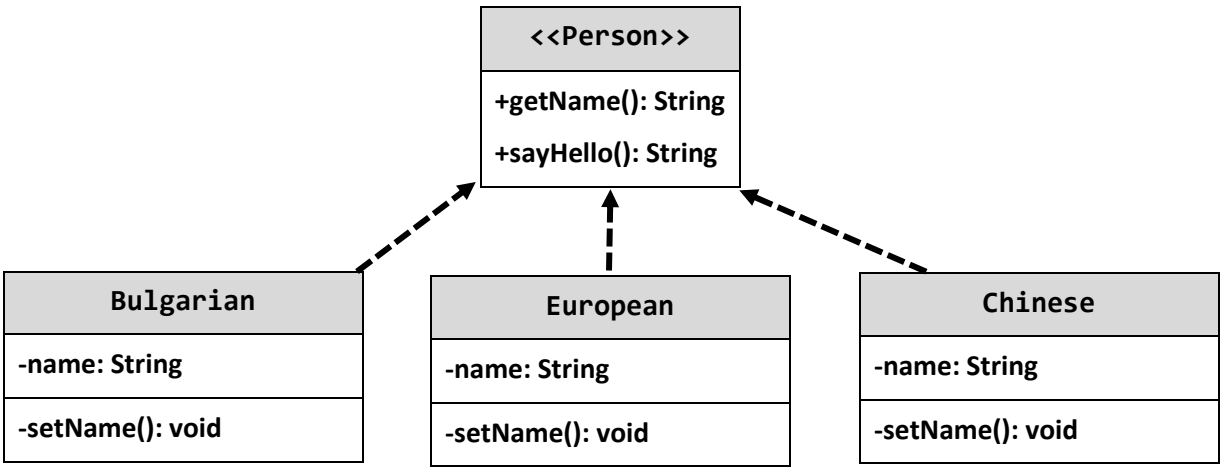
```
public static void main(String[] args) {
    Sellable seat = new Seat("Leon", "Gray", 110, "Spain", 11111.1);
    Rentable audi = new Audi("Leon", "Gray", 110, "Spain", 3, 99.9);

    printCarInfo(seat);
    printCarInfo(audi);
}

private static void printCarInfo(Car car) {
    System.out.println(String.format(
            "%s is %s color and have %s horse power",
            car.getModel(),
            car.getColor(),
            car.getHorsePower()));
    System.out.println(car.toString());
}
```

Examples

Input	Output
	Leon is gray and have 110 horse power This is Leon produced in Spain and have 4 tires

4. Say Hello

Build hierarchy from classes and interfaces for this UML diagram



Your hierarchy have to be used with this code

Main.java

```
public static void main(String[] args) {
    List<Person> persons = new ArrayList<>();
```

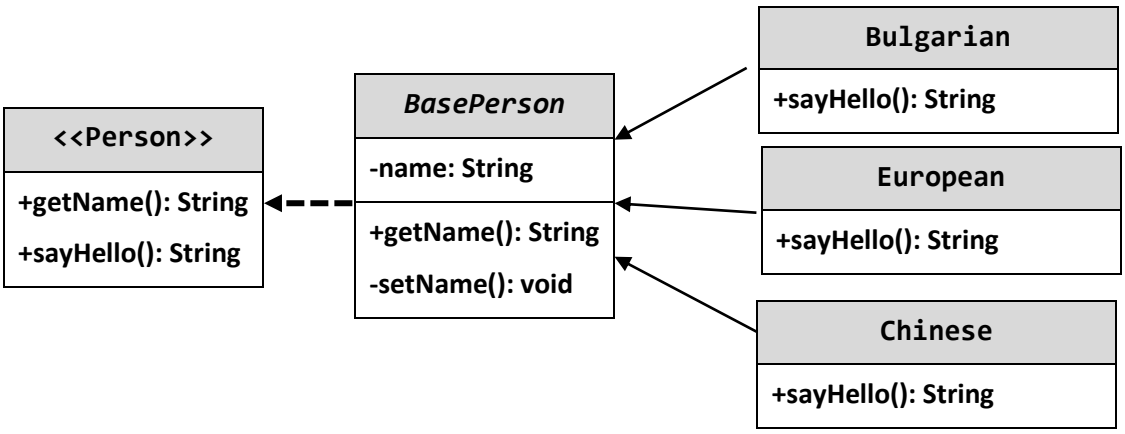
```
    persons.add(new Bulgarian("Pesho"));
    persons.add(new European("Pesho"));
    persons.add(new Chinese("Pesho"));

    for (Person person : persons) {
        print(person);
    }
}

private static void print(Person person) {
    person.sayHello();
}
```

Examples

Input	Output
	Здравей Hello Djydjybydjy

5. Say Hello Extend

Build hierarchy from classes and interfaces for this UML diagram



Your hierarchy have to be used with this code

Main.java

```
public static void main(String[] args) {
    List<Person> persons = new ArrayList<>();


    persons.add(new Bulgarian("Pesho"));
    persons.add(new European("Pesho"));
    persons.add(new Chinese("Pesho"));

    for (Person person : persons) {
        print(person);
    }
```

```
}

private static void print(Person person) {
    person.sayHello();
}
```

Examples

Input	Output
	Здравей Hello Djydjybydjy