Assignment 3

1. Explain polymorphism.
   Polymorphism is the ability of different objects to respond in a unique way to the same message. It can be represent many form in other situation or area.

2. What is overloading?
   Two or more methods may have the same name but have different parameters.

3. What is overriding?
   Overriding occurs when a subclass has the same method as the parent class. Method overriding provides a particular implementation of a method declared by one of its parent.

4. What does the final mean in this method: public void doSomething(**final** Car aCar){}
   It has a final parameter, it is a parameter that declared and cannot be changed.

5. Suppose in question 4, the Car class has a method setColor(Color color){…}, inside doSomething method, Can we call aCar.setColor(red);?
   No, since it is final parameter, it cannot changed to red.

6. Can we declare a static variable inside a method?
   We cannot declare static variable inside the method or constructor because static variable are class level variables.

7. What is the difference between interface and abstract class?
   An abstract class allows you to create functionality that subclasses can implement or override. An interface only allows you to define functionality, not implement it.

8. Can an abstract class be defined without any abstract methods?
   Yes, you can declare abstract class without defining an abstract method in it. Once you declare a class abstract it indicates that the class is incomplete. You will not able to instantiate it.

9. Since there is no way to create an object of abstract class, what's the point of constructors of abstract class?
   The purpose of the constructor is to initialize the newly created object. In abstract class, we have an instance variable, abstract methods, and non-abstract method. We need to initialize the non-abstract methods and instance variable. Therefore, abstract classes have a constructor.

10. What is a native method?
    A Java method that start in a language other than Java. They can access system specific functions and APIs that are not available directly in Java.

11. What is marker interface?
    A marker interface is an interface that has no methods or constants inside it. It provides run-time type information about objects. So the complier and JVM have additional information about the object.

12. Why to override equals and hashCode methods?
    We know that two objects are considered equal only if their references point to the same object, and unless we override equals and hashCode methods, the class object will not behave properly on hash-based collections like HashMap, HashSet, and Hashtable. This is because hash-based collections are organized like a sequence of buckets, and the hash code value of an object is used to determine the bucket where the object would be stored, and the same hash code is used again to find the object's position in the bucket.

13. What's the difference beween int and Integer?
    Int is primitive types of data in Java, and Integer is its wrapper class. It turns int to an objects, so we can use the wrapper class when dealing with Objects. And Integer has default value of "0". But int does not have a default value.

14. What is serialization?
    It means to convert its state to a byte stream so that the byte stream can be revered back into a copy of the object.

15. Create List and Map. List A contains 1,2,3,4,10(integer) . Map B contains ("a","1") ("b","2") ("c","10") (key = string, value = string)
    Question: get a list which contains all the elements in list A, but not in map B.

```java
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class ListAndMap {
    public static void main(String[] args) {
        List<Integer> list = new ArrayList<>();
        list.add(1);
        list.add(2);
        list.add(3);
        list.add(4);
        list.add(10);
        Map<String,String> map = new HashMap<>();
        map.put("a","1");
        map.put("b","2");
        map.put("c","10");
        List<Integer> result = new ArrayList<>();
        for (int i = 0; i < list.size(); i++) {
            int a = list.get(i);
            String b = Integer.toString(a);
            if (!map.containsValue(b)){
                result.add(a);
            }
        }
        System.out.println(result);
    }
}
```

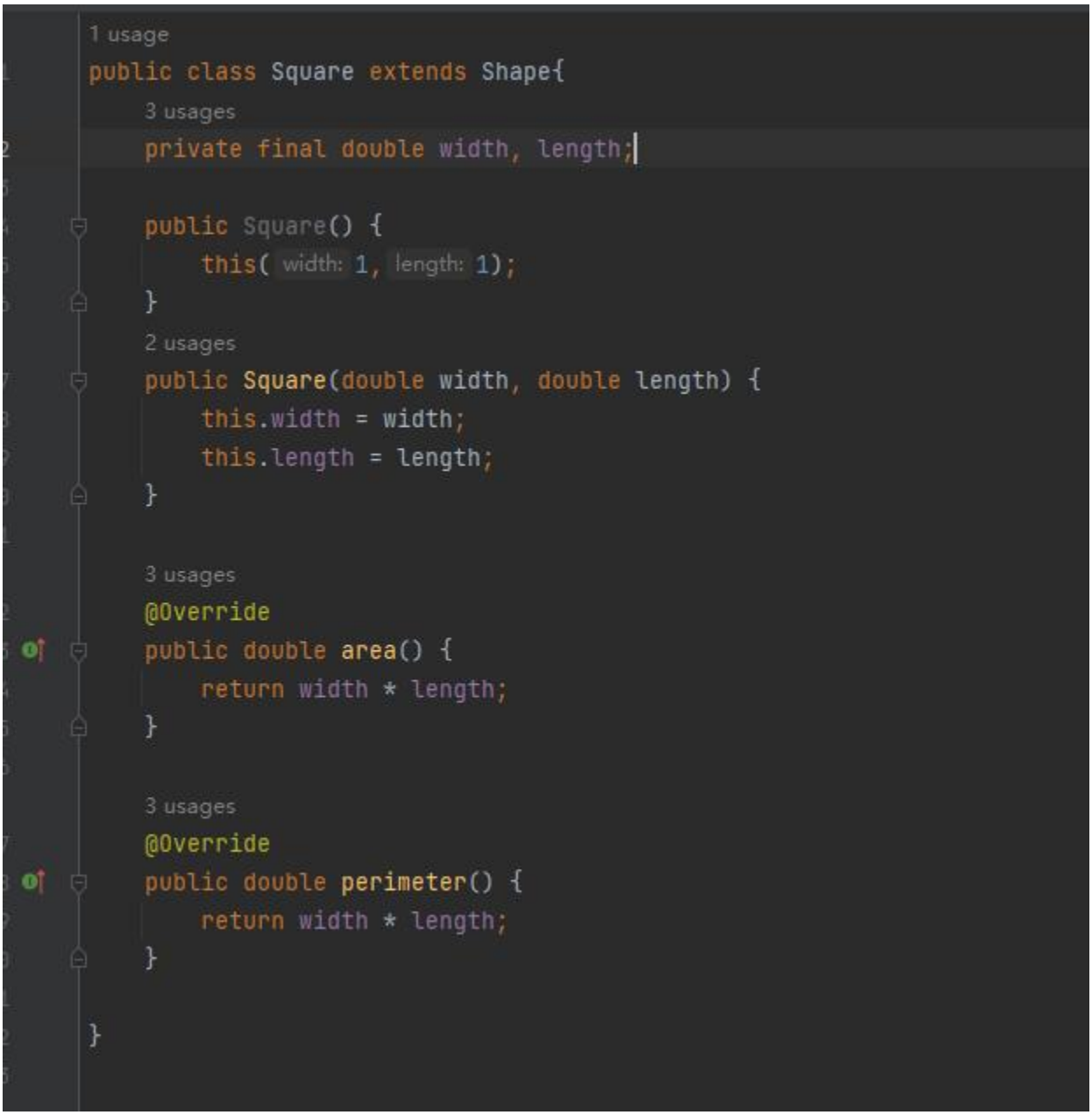
```
"C:\Program Files\Java\jdk-11.0.15.1\bin\java.exe" "-jav
[3, 4]

Process finished with exit code 0
```

16. Implement a group of classes that have common behavior/state as Shape. Create Circle, Rectangle and Square for now as later on we may need more shapes. They should have the ability to calculate the area. They should be able to compare using area. Please write a program to demonstrate the classes and comparison. You can use either abstract or interface. Comparator or Comparable interface.

```java
6 usages  3 inheritors
public abstract class Shape {
    3 usages  3 implementations
    public abstract double area();
    3 usages  3 implementations
    public abstract double perimeter();
}
```

```java
1 usage
public class Circle extends Shape{
    3 usages
    private final double radius;
    2 usages
    final double pi = Math.PI;

    public Circle() {
        this( radius: 1);
    }
    2 usages
    public Circle(double radius) {
        this.radius = radius;
    }

    3 usages
    @Override
    public double area() {
        // A = π r^2
        return pi * Math.pow(radius, 2);
    }

    3 usages
    public double perimeter() {
        // P = 2πr
        return 2 * pi * radius;
    }
}
```

```java
public class Rectangle extends Shape {
    3 usages
    private final double width, length;

    public Rectangle() {
        this( width: 1, length: 1);
    }
    1 usage
    public Rectangle(double width, double length) {
        this.width = width;
        this.length = length;
    }

    @Override
    public double area() {
        // A = w * l
        return width * length;
    }

    @Override
    public double perimeter() {
        // P = 2(w + l)
        return 2 * (width + length);
    }
}
```


1 usage
public class Square extends Shape{
3 usages
private final double width, length;
public Square() {
this(width: 1, length: 1);
}
2 usages
public Square(double width, double length) {
this.width = width;
this.length = length;
}
3 usages
@Override
public double area() {
return width * length;
}
3 usages
@Override
public double perimeter() {
return width * length;
}
}

```java
public class ShapeTest {
    public static void main(String[] args) {

        // Rectangle test
        double width = 5, length = 7;
        Shape rectangle = new Rectangle(width, length);
        System.out.println("Rectangle width: " + width + " and length: " + length
                + "\nResulting area: " + rectangle.area()
                + "\nResulting perimeter: " + rectangle.perimeter() + "\n");

        // Circle test
        double radius = 5;
        Shape circle = new Circle(radius);
        System.out.println("Circle radius: " + radius
                + "\nResulting Area: " + circle.area()
                + "\nResulting Perimeter: " + circle.perimeter() + "\n");

        // Square test
        double   w = 5, l = 5;
        Shape Square = new Square(w, l);
        System.out.println("Square width: " + w + " and length: " + l
                + "\nResulting area: " + Square.area()
                + "\nResulting perimeter: " + Square.perimeter() + "\n");
    }
}
```

```
Rectangle width: 5.0 and length: 7.0
Resulting area: 35.0
Resulting perimeter: 24.0

Circle radius: 5.0
Resulting Area: 78.53981633974483
Resulting Perimeter: 31.41592653589793

Square width: 5.0 and length: 5.0
Resulting area: 25.0
Resulting perimeter: 25.0


Process finished with exit code 0
```