Assignment 3

1. Explain polymorphism.
   Polymorphism is to perform a single action in different ways.

2. What is overloading?
   Overloading is to have multiple methods or constructor with the same name but different argument lists or return type.

3. What is overriding?
   Overriding is to implement the same method from the super class in a different way.

4. What does the final mean in this method: public void doSomething(**final** Car aCar){}
   This aCar variable can not be re-assigned to other object.

5. Suppose in question 4, the Car class has a method setColor(Color color){…}, inside doSomething method, Can we call aCar.setColor(red);?
   Yes, you can change the content of a final variable.

6. Can we declare a static variable inside a method?
   No, because static variable is a class level variable.

7. What is the difference between interface and abstract class?
   An interface is a 100% pure abstract class, abstract class could have concrete method but interface can not.
8. Can an abstract class be defined without any abstract methods?
   Yes
9. Since there is no way to create an object of abstract class, what's the point of constructors of abstract class?
   Child class could invoke the constructor in a parent abstract class.

10. What is a native method?
    A native method is a method implemented in languages other than java.

11. What is marker interface?
    Marker interface has no methods or constants inside it, it provides run-time type information about the object.

12. Why to override equals and hashCode methods?
    Because in hash-based collection, if those two methods are not overridden together, we will get different hashcode for the same object or vice versa which violates the contract for Obejct.hahsCode().

13. What's the difference between int and Integer?
    Int is a primitive data type but integer is a class which objects are created from.

14. What is serialization?
    Serialization converts an class's Object's state to a byte stream which could be reverted 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.

```
private List getList(List<Integer> list, Map<String,Integer> map) {
    Set s=(Set)map.entrySet();
    Iterator<Integer> ir=s.iterator();
    ArrayList<Integer> l=new ArrayList<>();
    while (ir.hasNext()) {
        l.add(ir.next());
    }
    List<Integer> res=new ArrayList<>();
    for (int i = 0; i < list.size(); i++) {
        if(!l.contains(list.get(i))) {
            res.add(list.get(i));
        }
    }
    return res;
}
```

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
public abstract class Shape implements Comparable<Shape> {
    abstract double area();
    @Override
    public int compareTo(Shape s) {
        double d=this.area()-s.area();
        int i=(int)d;
        return i;
    }
}
class Circle extends Shape{
    double radius;
    public Circle(double radius) {
        this.radius=radius;
    }
    @Override
    public double area() {
        return Math.PI*radius*radius;
    }

}
class Rectangle extends Shape{
    double width;
    double height;
    public Rectangle(double width,double height) {
        this.width=width;
        this.height=height;
    }
    @Override
    public double area() {
        return this.width*this.height;
    }
}
class Square extends Shape{
    double side;
    public Square(double side) {
        this.side=side;
    }
    @Override
    public double area() {
        return side*side;
    }
}
```