Inheritance, polymorphism Assignment 19th June

1. What is inheritance in java?
Ans: Deriving the properties and behaviors of one class into another class is known as inheritance.

2. What is superclass and subclass?
Ans: The class from which the properties are being derived is called as superclass and the class which is deriving another Class property is called as subclass.

3. How is inheritance implemented/achieved in java?
Ans: In java inheritance is achieved by using child class name, extends keyword followed by the parent class name.
For e.g. class Tiger extends Animal{}

4. What is Polymorphism?
Ans: When one method or variable shows more than one form then it is known as polymorphism.

5. Differentiate between method overloading and overriding?
Ans: Method overloading:
1. The process of extending the existing method functionality is called Method Overloading.
2. In method overloading, different method signatures must be provided to the methods.
3. With or without inheritance, method overloading can be performed.

Method overriding:
1. The process of replacing existing method functionality with new functionality is called method overriding.
2. In method overriding, same method prototypes must be provided to the methods.
3. Method overriding can only be performed by using inheritance.

6. What is an abstraction explain with an example?
Ans: Abstraction is a process of hiding the implementation details and showing only functionality to the user.

```
Eg., abstract class Animal{
    abstract void eat();
}
class Tiger extends Animal{
    void eat(){
        System.out.println("Tiger can eat");
    }
    void run(){
        System.out.println("tiger can run");
    }
}
public class AbstractionEx {
    public static void main(String[] args ) {
        Animal a= new Tiger();
        a.eat();
    }
}
```

7. What is difference between an abstract method and final method in java? Explain with an example.
Ans: Abstract method:
If a method is declared as abstract means it has no body implementation and it needs to be implement in child class.
Final method:
If a method declared as final than that method cannot be overridden only accessible by the reference variable.

```
    abstract class Animal1{
    abstract void eat();
   final public void sleep(){
        System.out.println("animal needs sleep");
    }
}
class Lion extends Animal1{
    public void eat(){
        System.out.println("Lion is eating");
    }
    //public void sleep()                    cannot be overridden because of final method.
}
public class AbsAndFinal {
    public static void main(String[] args) {
        Animal1 a = new Lion();
        a.sleep();
    }
}
```

8. What is the final class in java?

Ans: If a class is declared as final, then the class won't participate in inheritance, if we try to do so then it would result in "CompileTimeError".

9. Differentiate between abstraction and encapsulation.

Ans: Abstraction:

Abstraction is a method to hide unwanted information we can implement abstraction by using abstract class.

Encapsulation:

Encapsulation is the method to hide data in single entity or unit along with a method to protect.
We can implement Encapsulation by using private, protected and public keyword.

10. Difference between Runtime and Compile time Polymorphism explain with an example?

Ans: 1.Runtime polymorphism:

Runtime polymorphism is achieved through method overriding. Where a method gets implemented in subclass which is declared in superclass.

```
Eg. class Animal2{
    public void eat(){
        System.out.println("Animal can eat");
    }
}
class Dog extends Animal2{
   // method overriding
    public void eat(){
        System.out.println("Dog is eating");
    }
}
public class RuntimeEx {
    public static void main(String[] args) {
        Dog d = new Dog();
        d.eat();
    }
}
```

2. Compile time polymorphism:

Compile time polymorphism is achieved through method overloading. Where multiple methods have same name but different type and number of parameters.

```
Eg. class Calc1 {
    public int add(int a, int b) {
        int result = a + b;
        return result;
    }
    public int add(int a, int b, int c) {
        int result = a + b + c;
        return result;
    }
}
public class CompileTimeEx {
    public static void main(String[] args) {
        Calc1 c = new Calc1();
        c.add(5,4);              // output 9
        c.add(3,4,5);         // output 12
    }
}
```