FULL STACK DEVELOPMENT – WORKSHEET 4

Q1.Write in brief about OOPS Concept in java with Examples. (In your own words)
Ans: This are some basic terms and concepts of OOPS in java:

1. **Object:**
   An object is a software bundle of related state and behavior. Software objects are often used to model the real-world objects that you find in everyday life. This lesson explains how state and behavior are represented within an object, introduces the concept of data encapsulation, and explains the benefits of designing your software in this manner.
2. **Class:**
   A class is a blueprint or prototype from which objects are created. This section defines a class that models the state and behavior of a real-world object. It intentionally focuses on the basics, showing how even a simple class can cleanly model state and behavior.
3. **Inheritence:**
   Inheritance provides a powerful and natural mechanism for organizing and structuring your software. This section explains how classes inherit state and behavior from their superclasses, and explains how to derive one class from another using the simple syntax provided by the Java programming language.
4. **Interface:**
   An interface is a contract between a class and the outside world. When a class implements an interface, it promises to provide the behavior published by that interface. This section defines a simple interface and explains the necessary changes for any class that implements it.
5. **Package:**
   A package is a namespace for organizing classes and interfaces in a logical manner. Placing your code into packages makes large software projects easier to manage. This section explains why this is useful, and introduces you to the Application Programming Interface (API) provided by the Java platform.
6. **Encapsulation:**
   It is a process of wrapping code and data together into a single unit, for example, a capsule which is mixed of several medicines. We can create a fully encapsulated class in Java by making all the data members of the class private. Now we can use setter and getter methods to set and get the data in it. Java Bean class is the example of a fully encapsulated class.
7. **Polymorphism:**
   It is a concept by which we can perform a single action in different ways, We base class object which is referenced by its subclass, so that this object can be restricted to use only subclass properties. If we overload a static method in Java, it is the example of compile time polymorphism. Runtime polymorphism or Dynamic Method Dispatch is a process in which a call to an overridden method is resolved at runtime rather than compile-time.
8. **Abstraction:**
   It is a process of hiding the implementation details and showing only functionality to the user. Another way, it shows only essential things to the user and hides the internal details. For example: Switch board, only shows button hide wirings.

Q2.Write simple programs (wherever applicable) for every example given in Answer 2.
Ans: This programs are also shows class object and package as well.
Program to show Inheritence:

```
package com.JavaBasics;
//Creating base/parent class

import java.sql.SQLOutput;

class Base {
    public int x;

    public int getX() {
        return x;
    }

    public void setX(int x) {
        System.out.println("I am in base class and setting value of x");
        this.x = x;
    }

    public void printMe() {
        System.out.println("Hi! this is method in base class");
    }
}


class Derived extends Base {
    public int y;
    public int getY() {
        return y;
    }
    public void setY(int y) {
        this.y=y;
    }
}

class Animal {
    public String color;
    public int age;


    public void setColor(String color) {
        this.color =color;
    }
    public String getColor() {
        System.out.println("The color of this animal is: ");
        return color;
    }

    public void setAge(int age) {
        this.age=age;
    }
    public int getAge() {
        System.out.println("The Age of animal is: ");
        return age;
    }
}

class Dog extends Animal {
    public String voice;
```

```
    public void setVoice() {
        System.out.println("Dog Barks");
    }

    public void getLegs() {
        System.out.println("Dog has 4 legs");
    }
}
public class ch10_Inheritance {
    public static void main(String[] args) {
        //calling base object

        Base b=new Base();
        b.setX(23);
        System.out.println("Value of x is:" + b.getX());

        Derived d=new Derived();
        d.setX(45);
        System.out.println("The value of inherit x from derived class is:
"+ d.getX());


        Animal a=new Animal();
        a.setAge(5);
        a.getAge();
        a.setColor("Brown");
        a.getColor();
    }
}
```

Program to show Polymorphism

```
package com.JavaBasics;
interface gps {
    void shareLocation();
    void trackLocation();
}
interface camera2 {
    void takeShot();
    void recordVideo();
}
interface mediaPlayer {
    void playSong();
    void createPlaylist();
}
class LatestSmartPhone implements gps, camera2, mediaPlayer {
    public void shareLocation() {
        System.out.println("Live Location Shared for 8 hours...");
    }
    public void trackLocation() {
        System.out.println("Given Location Tracking starts, follow the
direction...");
    }
    public void takeShot() {
        System.out.println("Captured the shot...");
    }
    public void recordVideo() {
        System.out.println("Recording in Progress...");
    }
    public void playSong() {
```

```
        System.out.println("Selected song starts playing");
    }
    public void createPlaylist() {
        System.out.println("Playlist of selected songs have been
created...");
    }
    public void classMethod() {
        System.out.println("This is method of LatestSmartPhone Class not
from any Interface");
    }
}

public class ch11_PolymorphismInInterface {
    public static void main(String[] args) {
        camera2 cam= new LatestSmartPhone();  //THis is a smartphone please
use it as a camera
        System.out.println("outputs of cam object , reference camera
interface and class object");
        cam.takeShot();
        cam.recordVideo();    //This two are valid for camera reference
object

       // cam.shareLocation();       gps method for camera reference are
not valid
       // cam.trackLocation();


        //cam.classMethod();           Even class methods are also not
allowed

        LatestSmartPhone lsp=new LatestSmartPhone();
        System.out.println("Outputs of lsp object, reference and object by
class ");
        lsp.shareLocation();
        lsp.trackLocation();
        lsp.takeShot();
        lsp.recordVideo();
        lsp.playSong();
        lsp.createPlaylist();

        System.out.println("Class methods on lsp object");
        lsp.classMethod();

    }
}
```

Program to show Abstraction:

```
package com.JavaBasics;

abstract class Parent3{
    public Parent3() {
        System.out.println("I am constructor of Base2 Class");
    }
    public void sayHello() {
        System.out.println("Hello Abhishek");
    }
    abstract public void greet();
    abstract public void greet2();
```

```
}
class Child2 extends Parent3{
    @Override
    public void greet() {
        System.out.println("Wahe guru");
    }
    @Override
    public void greet2() {
        System.out.println("Sat Shree Akal");
    }
}
abstract class Child3 extends Parent3 {
    public void read() {
        System.out.println("Reading Novel");
    }
}

public class ch11_AbstractClass {
    public static void main(String[] args) {

//        Abstract classes cannot be instantiated

//        Parent3 p=new Parent3() ;

        Child2 c2=new Child2();

//        Child3 c3= new Child3();  Abstract class cannot be instantiate

    }
}
```

Multiple Choice Questions

Q1. Which of the following is used to make an Abstract class?
A. Making at least one member function as pure virtual function
B. Making at least one member function as virtual function
C. Declaring as Abstract class using virtual keyword
D. Declaring as Abstract class using static keyword

Ans: D Declaring as Abstract class using static keyword.

Q2. Which of the following is true about interfaces in java.
1) An interface can contain the following type of members.
....public, static, final fields (i.e., constants)
....default and static methods with bodies
2) An instance of the interface can be created.
3) A class can implement multiple interfaces.
4) Many classes can implement the same interface.
A. 1, 3 and 4
B. 1, 2 and 4
C. 2, 3 and 4
D. 1,2,3 and 4

Ans: A, option 2 is incorrect as we can not create instance of interface.

Q3. When does method overloading is determined?
A. At run time
B. At compile time
C. At coding time
D. At execution time

Ans: B At compile time. Also termed as compile time polymorphism.

Q4.What is the number of parameters that a default constructor requires?
A. 0
B. 1
C. 2
D. 3

Ans: A 0 as constructor can be initialize without parameter as well.

Q5. To access data members of a class, which of the following is used?
A. Dot Operator
B. Arrow Operator
C. A and B both as required
D. Direct call

Ans: A. Dot Operator

Q6.Objects are the variables of the type ____?
A. String
B. Boolean
C. Class
D. All data types can be included

Ans: D Objects can be of any type.

Q7. A non-member function cannot access which data of the class?
A. Private data
B. Public data
C. Protected data
D. All of the above

Ans: B Private data

Q8. Predict the output of following Java program

```
class Test {
int i;
}
class Main {
public static void main(String args[]) {
Test t = new Test();
```

```
System.out.println(t.i);
}
}
```
A. garbage value
B. 0
C. compiler error
D. runtime Error

Ans: B 0

Q9.Which of the following is/are true about packages in Java?
1) Every class is part of some package.
2) All classes in a file are part of the same package.
3) If no package is specified, the classes in the file
go into a special unnamed package
4) If no package is specified, a new package is created with
folder name of class and the class is put in this package.
A. Only 1, 2 and 3
B. Only 1, 2 and 4
C. Only 4
D. Only 1, 3 and 4

Ans: A 4th statement is incorrect.

For Q10 to Q25 find output with explanation.

Q10.Predict the Output of following Java Program.
```
class Base {
public void show() {
System.out.println("Base::show() called");
}
}
class Derived extends Base {
public void show() {
System.out.println("Derived::show() called");
}
}
public class Main {
public static void main(String[] args) {
Base b = new Derived();;
b.show();
}
}
```

Ans: Derived::show() called
As b.show will execute the show method of derived class which commands to print this line. Here b is the element reference by base class variable but object of derived class, hence derived class method will execute.

Q11. What is the output of the below Java program?

```
class Base {
final public void show() {
System.out.println("Base::show() called");
}
}
class Derived extends Base {
public void show() {
System.out.println("Derived::show() called");
}
}
class Main {
public static void main(String[] args) {
Base b = new Derived();;
b.show();
}
}
```

Ans: Here this program will show error, As the overridden method show() is final and hence derived class can't override this now.

Q12.Find output of the program.

```
class Base {
public static void show() {
System.out.println("Base::show() called");
}
}
class Derived extends Base {
public static void show() {
System.out.println("Derived::show() called");
}
}
class Main {
public static void main(String[] args) {
Base b = new Derived();
b.show();
}
}
```

Ans: Base::show() called

As the base method is static in this case, Base class show method will invoke

Q13.What is the output of the following program?

```
class Derived
{
public void getDetails()
```

```
{
System.out.printf("Derived class ");
}
}
public class Test extends Derived
{
public void getDetails()
{
System.out.printf("Test class ");
super.getDetails();
}
}
public static void main(String[] args)
{
Derived obj = new Test();
obj.getDetails();
}
}
```

Ans: Shows an error: class Test is public, should be declared in a file named Test.java

After removing public keyword,

Output : Test class Derived class

Q14. What is the output of the following program?

```
class Derived
{
public void getDetails(String temp)
{
System.out.println("Derived class " + temp);
}
}
public class Test extends Derived
{
public int getDetails(String temp)
{
System.out.println("Test class " + temp);
return 0;
}
public static void main(String[] args)
{
Test obj = new Test();
obj.getDetails("Name");
}
}
```

Ans: output: Test class Name

Although I have to change the return type of test class getDetail method to void.

Q15.What will be the output of the following Java program?

```
class test
{
public static int y = 0;
}
class HasStatic
{
private static int x = 100;
public static void main(String[] args)
{
HasStatic hs1 = new HasStatic();
hs1.x++;
HasStatic hs2 = new HasStatic();
hs2.x++;
hs1 = new HasStatic();
hs1.x++;
HasStatic.x++;
System.out.println("Adding to 100, x = " + x);
test t1 = new test();
t1.y++;
test t2 = new test();
t2.y++;
t1 = new test();
t1.y++;
System.out.print("Adding to 0, ");
System.out.println("y = " + t1.y + " " + t2.y + " " + test.y);
}
}
```

Ans: Output: **Adding to 100, x = 104**
Adding to 0, y = 3 3 3

Q16.Predict the output

```
class San
{
public void m1 (int i,float f)
{
System.out.println(" int float method");
}
public void m1(float f,int i);
{
System.out.println("float int method");
}
public static void main(String[]args)
{
San s=new San();
s.m1(20,20);
}  }
```

Ans: reference to m1 is ambiguous, both method m1(int,float) in com.InternshipFlipRobo.San and method m1(float,int) matches

Q17.What is the output of the following program?

```
public class Test
{
public static void main(String[] args)
{
int temp = null;
Integer data = null;
System.out.println(temp + " " + data);
}
}
```

Ans: incompatible types: <nulltype> cannot be converted to int , Data type error

Q18.Find output

```
class Test {
protected int x, y;
}
class Main {
public static void main(String args[]) {
Test t = new Test();
System.out.println(t.x + " " + t.y);
}
}
```

Ans: 0 0, Both x and y default set to 0

Q19. Find output

```
// filename: Test2.java
class Test1 {
Test1(int x)
{
System.out.println("Constructor called " + x);
}
}
class Test2 {
Test1 t1 = new Test1(10);
Test2(int i) { t1 = new Test1(i); }
public static void main(String[] args)
{
Test2 t2 = new Test2(5);
}
}
```

Ans: output: Constructor called 10
Constructor called 5

First Test1 constructor with value of i=5 will invoke, later with i=10;

Q20.What will be the output of the following Java program?

```
class Main
{
public static void main(String[] args)
{
int []x[] = {{1,2}, {3,4,5}, {6,7,8,9}};
int [][]y = x;
System.out.println(y[2][1]);
}
}
```

Ans: 7 , Array element of x array corresponding to (2,1) will invoke.

Q21.What will be the output of the following Java program?

```
class A
{
int i;
public void display()
{
System.out.println(i);
}
}
class B extends A
{
int j;
public void display()
{
System.out.println(j);
}
}
class Dynamic_dispatch
{
public static void main(String args[])
{
B obj2 = new B();
obj2.i = 1;
obj2.j = 2;
A r;
r = obj2;
r.display();
}
}
```

Ans: 2 , As value corresponds to obj2.i will copied in r (object of class A) . Hence, display method of class A will invoke. And print I as 2.

Q22. What will be the output of the following Java code?

```
class A
{
int i;
void display()
```

```
{
System.out.println(i);
}
}
class B extends A
{
int j;
void display()
{
System.out.println(j);
}
}
class method_overriding
{
public static void main(String args[])
{
B obj = new B();
obj.i=1;
obj.j=2;
obj.display();
}
}
```

Ans: 2, As obj is object of class B, so display method of class B will invoke , hence print value of j.

Q23.What will be the output of the following Java code?

```
class A
{
public int i;
protected int j;
}
class B extends A
{
int j;
void display()
{
super.j = 3;
System.out.println(i + " " + j);
}
}
class Output
{
public static void main(String args[])
{
B obj = new B();
obj.i=1;
obj.j=2;
obj.display();
}
```

```
}
```

Ans: 1 2 , As display method of class B will invoke which sets variable j of superclass A as 3 but will print i and j of B class, which is 1 and 2.

Q24.What will be the output of the following Java program?

```
class A
{
public int i;
public int j;
A()
{
i = 1;
j = 2;
}
}
class B extends A
{
int a;
B()
{
super();
}
}
class super_use
{
public static void main(String args[])
{
B obj = new B();
System.out.println(obj.i + " " + obj.j)
}
}
```

Ans: 1 2 , with object creation of Class B , constructor of this class will automatically invoke, which has super keyword, which invoke Base Class A, hence set i=1 and j= 2 , and later command print obj.i and obj.j.

Q 25. Find the output of the following program.

```
class Test
{
int a = 1;
int b = 2;
Test func(Test obj)
{
Test obj3 = new Test();
obj3 = obj;
obj3.a = obj.a++ + ++obj.b;
obj.b = obj.b;
return obj3;
}
public static void main(String[] args)
```

```
{
Test obj1 = new Test();
Test obj2 = obj1.func(obj1);
System.out.println("obj1.a = " + obj1.a + " obj1.b = " + obj1.b);
System.out.println("obj2.a = " + obj2.a + " obj1.b = " + obj2.b);
}
}
```

Ans: Output **obj1.a = 4 obj1.b = 3**
obj2.a = 4 obj1.b = 3

As, Object 1 and object 2 refers to same memory address, they will have same values.