FULL STACK DEVELOPMENT - WORKSHEET 4

1. Write in brief about OOPS Concept in java with Examples.

Ans. Object-Oriented Programming (OOP) is a programming paradigm that revolves around the concept of objects, which are instances of classes. Java is an object-oriented programming language, and it incorporates several key principles of OOP.

Classes and Objects: In Java, a class is a blueprint or template that defines the properties (attributes) and behaviors (methods) of objects. An object, on the other hand, is an instance of a class. For example, consider a class called "Car" that represents cars. Each individual car, such as "BMW" or "Toyota," can be considered as an object of the Car class.

Encapsulation: Encapsulation is the process of hiding internal details and providing access to data and methods through a well-defined interface. In Java, this is achieved by using access modifiers (e.g., private, protected, public) to control the visibility of class members. For instance, in a Car class, you can encapsulate the speed attribute by making it private and providing a public method to set or retrieve its value.

```
public class Car {
	private int speed;

	public void setSpeed(int speed) {
		this.speed = speed;
	}
	public int getSpeed() {
		return speed;
	}
}
```

Inheritance: Inheritance allows classes to inherit the properties and behaviors of other classes. In Java, a class can extend only one parent class, but it can implement multiple interfaces. This promotes code reuse and hierarchical organization of classes. Let's say we have a class called "SportsCar" that extends the Car class. The SportsCar class will inherit the attributes and methods of the Car class.

```
public class SportsCar extends Car {
	// Additional properties and methods specific to the Sports Car.
```

Polymorphism: Polymorphism means the ability of an object to take on many forms. In Java, polymorphism is achieved through method overriding and method overloading. Method overriding allows a subclass to provide a different implementation of a method defined in its superclass. Method overloading allows a class to have multiple methods with the same name but different parameters.

```
public class Animal {
        public  void makeSound() {
                System.out.println("Animal is making some sound");
        }
}

public class Cat extends Animal {
        @override
        public void makeSound() {
                System.out.println("meaw!");
        }
}

public class Dog extends Animal {
        @override
        public void makeSound() {
                System.out.println("Woof!");
        }
}
```

Abstraction: Abstraction refers to the process of hiding complex implementation details and providing a simplified interface. In Java, abstraction can be achieved through abstract classes and interfaces. Abstract classes cannot be instantiated but can be subclassed, while interfaces define a contract that implementing classes must adhere to.

```
public abstract class Shape {
        public abstract void draw();
}
public class Circle extends Shape {
        @override
        public void draw() {
                System.out.println("Drawing a circle.");
        }
}
```

Q1. Which of the following is used to make an Abstract class?

Ans. C. Declaring as Abstract class using virtual 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.

Ans. A. 1, 3 and 4

Q3. When does method overloading is determined?

Ans. B. At compile time.

Q4.What is the number of parameters that a default constructor requires?

Ans A. 0 *(A default constructor is a constructor that is automatically provided by the Java compiler if no explicit constructor is defined in a class. It does not take any parameters.)*

Q5.To access data members of a class, which of the following is used?

Ans. A. Dot Operator. *(In Java, the dot operator (.) is used to access and refer to the data members (fields) and methods of a class. It is used to access variables and invoke methods on objects of the class)*

Q6. Objects are the variables of the type ____?

Ans. C. Class.

Q7.A non-member function cannot access which data of the class?

Ans. A. 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);
}
}
```

Ans. B. 0 (In Java the default value of int is zero(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.

Ans. A. Only 1, 2 and 3

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. The output of the program will be: “Derived::show() called”

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. Compilation error *(a final method in the Base Class cannot be overridden)* .

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 *(an Object is created for Base class, therefore it calls the show() method from the Base Class)*

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. The output is: "Test class Derived class"

(an Object is created for Derived class, which further calls getDetails() method and prints ***"Test class"*** *with in this method it calls another method pointing to the Base class. Therefore the final output is* ***"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. The program throws an error during compilation. The method declaration in Test class is incorrect.

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. **Adding to 100, x = 104** *(variable x is incremented four times hence 104)*

Adding to 0, y = 3 3 3 *(variable y is incremented three times and the final value before printing is 3)*

Q16.Predict the output

```
class San {
      public void m1 (int i,float f) {
            System.out.println(" int float method");
            }

      // method m1 is wrongly terminated with a semi-colon.
      public void m1(float f,int i); {
            System.out.println("float int method");
}
public static void main(String[]args) {
San s=new San();

// the object  ‘s’ gets confused to call which m1 method, as both
methods are passed with same type of/ number of parameters
s.m1(20,20);
}
}
```

Ans. **Compilation error.**

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 Compilation error. *("int" is a primitive data type and cannot hold a "null" value)*

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. Output: "0 0" *(default value of int is '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. Constructor called 10 *(When Test2 is called the parameter 'x' is passed with value 10)*
Constructor called 5 *(When Test1 is called the parameter 'x' is passed with value 5)*

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 *(a Two dimensional array pointing to the second element in the third row)*

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 *(r is a reference variable for class 'A' which is pointing to the object of class 'B'. when the display() method is called with 'r' reference it indirectly points to obj2.j)*

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 *(here the object is created for class 'B'. when display() method is called it prints the value assigned to the variable '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 3 *(a protected variable can also be accessed from the extended class. When the object 'B' calls the display() method it prints 'i' & 'j' variable values. 'i' value is 1 where as super.j value is printed.)*

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 *(the Object for class 'B' is calls method super() which points to the Base class and prints the values 1 & 2)*

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. **obj1.a = 4 obj1.b = 3**
obj2.a = 4 obj1.b = 3
(a++ + ++b = 1 + 3(pre-increment) => 4)
(after increment the 'b' value is updated to 3)