Disha Agarwal
23WU0102052

OOPS ASSIGNMENT

Q1) Define each with examples.

1) Classes :- A class is a set of objects which share common characteristics and attributes. It is user-defined blueprint or prototype from which objects are created.

Eg :- Student is a class while a particular student called anvi is an object.

2) Objects :- An object is a basic unit of object-oriented programming and represents real-life entities. Objects are instances of a class that are created to use attributes and methods of a class.

Eg :- Dog

Identity	State/Attributes ~~Breed Age~~	Behaviours
Name of the dog	Breed, Age, Colour	Bark, Sleep, Eat.

Interfacing :- It is sending messages with them, i.e., send messages to them, receive messages from them and perhaps make the different objects composing the class interact with each other.

Q2) Explain the four fundamental pillars of OOPS.

A) The four fundamental pillars are :-

1) Encapsulation :- It is the process of wrapping code and data together into a single unit. The data is unaccessable to the outside world and only methods wrapped in class can access it.

2) Abstraction :- It refers to the act of representing essential features without including the background details or explanations.

3) Inheritance :- It is a mechanism of deriving one class from another. It allows the derived class to inherit features from parent class.

4) Polymorphysm :- It is the ability to create a function, variable or an object that has more than 1 form. The concept is expressed by 'one interface, many methods'.

Q3) Explain in brief the different access modifiers used in Java.

A) We can make any variable, class or data member private by using the private keyword. It's access is within class only. This is known as private access modifier.

Eg:- public package 1;
public class classB;
private void print() {
System.out.println("Hello World");
}
}

Protected Access Modifier :- We can make any variable, class or data member protected by using the protected keyword. It's access is same packages or subclass.

Eg:- package package2;
public class classC{
protected void print() {
System.out.println("Hello World");
}
}

Public Access Modifier :- We can make any variable, class or data member public by using the 'public' keyword. It can be accessed from anywhere in a project.

Eg:- package package2;
public class classC {
public void print() {
System.out.println("Hello World");
}
}

Default Access Modifier :- When you don't put any other modifiers with any variable, class or datatype then it is default access modifier. It can be accessed through same package only.

Eg :-
```
package package2;
public class class c{
    void print() {
        System.out.println("Hello World");
    }
}
```

9) Write a brief summary for different kinds of variables along with differences.

1) Local Variable :- A variable which is declared inside the method.

2) Instance Variable :- A variable which is declared inside class but outside method.
- It is not declared static.

3) Static Variable :- A variable that is declared static are static variables.
- It cannot be local.

	Instance Variable	Static Variable
1)	Each object will have its own copy of instance variable.	We can only have one copy of static variable per class no matter the no. of objects.
2)	Changes made in instance variable using one object will not be reflected in other objects as each has its own copy.	Changes will be reflected in other objects are static variables are common to all objects of a class.

3)	We can access them through object references.
	Eg:- Class Sample
	{
	int a;
	}
		We can access them through using class name.
		Eg:- Class Sample
		{ static int a;
		}

4)	2 bytes, 4 bytes, 8 bytes	Inheritance, Abstraction

9) Write the output for the following code.

1)
```
import java.io.*;
import java.util.*;
class GFG {
    public static void main (String [] args) {
        // Divident
        int a = 15
```

```
        // Divisor
          int b = 8
        // Mod
          int k = a % b
          System.out.println(k)
      }
  }
```

Output :- 1

2)
```
import java.io.*;
class Ternary {
    public static void main(String[] args) {
        int n1=5, n2=10, res;
        System.out.println("First num:", +n1);
        System.out.println("Second num:" + n2);
        res = (n1>n2) ? (n1+n2) : (n1-n2);
        System.out.println("Result :" + res);
    }
}
```

Output :

First num : 5

Second num : 10

Result : -5

9)
```
public class operator3 {
    public static void main (String [] args) {
        int a = 10;
        int b = 5;
        int c = 20;
        System.out.println(a<b      a++<c);
        System.out.println(a);
        System.out.println(a<b  a++<c);
        System.out.println(a);
    }
}
```

Output:

False
10
False
11

9)
```
public class Nested for Example {
    public static void main (String [] args) {
        for (int i = 1; i<=3; i++) {
            for (int i = 1; j<=3; j++) {
                System.out.println (i+" "+j);
    }}}}
```

Output :-

```
1 1
1 2
1 3
2 1
2 2
2 3
3 1
3 2
3 3
```

3)
```
public class operator Example {
    public static void main ( String [] args) {
        int n = 10
        System . out . println (n++)
        System . out . println ( ++n)
        System . out . println ( n --)
        System . out . println (-- n)
    }
}
```

Output :

10
12
12
10

4)
```
public class operator Example {
    public static void main (String [] args) {
        int a = 10 ;
        int b = 10 ;
        System . out . println (a++ + ++a) ; // 10 + 12 = 22
        System . out . println (b++ + . ) ; // 10+11 = 21
    }
}
```

Output :

22
21

9) public class operator3 {

```
    public static void main (String [] args) {
        int a = 10;
        int b = 5;
        int c = 20;
        System.out.println (a<b     a++<c);
        System.out.println (a);
        System.out.println(a<b  a++ <c);
        System.out.println(a);
    }
}
```

Output:

False

10

False

11

9) public class Nested for Example {

```
    public static void main (String [] args) {
        for (int i = 1; i<=3; i++) {
            for (int i = 1; j<=3; j++) {
                System.out.println (i+" "+j);
            }
        }
    }
}
```

Output :-

```
1 1
1 2
1 3
2 1
2 2
2 3
3 1
3 2
3 3
```

8) Fill in the blanks.

1) The default size for short, int, long datatype is 2-bytes, 4-bytes, 8-bytes.

2) During Inheritance, the derived members of field and methods from the parent class can be easily accessible from the subclass.

3) Inheritance is a property of OOPS in which member function and data members of class can be used by another class.

4) Abstraction hides the complexity by giving you a more abstraction picture of a complex system.

9) ~~Output~~