NAME : MUHAMMAD SAEED KHAN

ID: 16015

SUBJECT : OOP

DEPARTMENT: BSCS-2

QUESTION NUMBER 1: a. Why access modifiers are used in java, explain in detail Private and Default access modifiers?

b. Write a specific program of the above mentioned access modifiers in java.

ANSWER: ACCESS MODIFIERS IN JAVA: (PART A and B are solved together)

As the name suggests access modifiers in Java helps to restrict the scope of a class, constructor , variable , method or data member. There are four types of access modifiers available in java:

Default – No keyword required
Private
Protected
Public

DEFAULT:
When no access modifier is specified for a class , method or data member – It is said to be having the default access modifier by default.

The data members, class or methods which are not declared using any access modifiers i.e. having default access modifier are accessible only within the same package.

In this example, we will create two packages and the classes in the packages will be having the default access modifiers and we will try to access a class from one package from a class of second package

```
//Java program to illustrate default modifier

package p1;


//Class Geeks is having Default access modifier

class Geek

{

   void display()

      {

          System.out.println("Hello World!");

      }

}

//Java program to illustrate error while

//using class from different package with

//default modifier

package p2;

import p1.*;


//This class is having default access modifier

class GeekNew
```

```
{
   public static void main(String args[])
     {
       //accessing class Geek from package p1
       Geeks obj = new Geek();


       obj.display();
     }
}
```

OUTPUT:

Compile time error.

PRIVATE:

The private access modifier is specified using the keyword private.

- The methods or data members declared as private are accessible only within the class in which they are declared.
- Any other class of same package will not be able to access these members.
- Top level Classes or interface can not be declared as private because

1- private means "only visible within the enclosing class".
2- protected means "only visible within the enclosing class and any subclasses"

Hence these modifiers in terms of application to classes, they apply only to nested classes and not on top level classes

In this example, we will create two classes A and B within same package p1. We will declare a method in class A as private and try to access this method from class B and see the result.

```
//Java program to illustrate error while

//using class from different package with

//private modifier

package p1;


class A

{

  private void display()

   {

      System.out.println("GeeksforGeeks");

   }

}


class B

{

  public static void main(String args[])

    {

        A obj = new A();

        //trying to access private method of another class

        obj.display();

    }

}
```

OUTPUT:

```
error: display() has private access in A
        obj.display();
```

QUESTION NUMBER 2: a. Explain in detail Public and Protected access modifiers?

b. Write a specific program of the above mentioned access modifiers in java.

ANSWER:

DETAIL EXPLAINATION WITH PROGRAMS:

PUBLIC:

The public access modifier is specified using the keyword public.

- The public access modifier has the widest scope among all other access modifiers.
- Classes, methods or data members which are declared as public are accessible from every where in the program. There is no restriction on the scope of a public data members

PROGRAM:

```
//Java program to illustrate

//public modifier

package p1;

public class A

{

  public void display()
```

```
      {
         System.out.println("GeeksforGeeks");
      }
}
package p2;
import p1.*;
class B
{
   public static void main(String args[])
      {
         A obj = new A;
         obj.display();
      }
}
```

OUTPUT:

```
GeeksforGeeks.
```

PROTECTED:

The protected access modifier is specified using the keyword protected

- The methods or data members declared as protected are accessible within same package or sub classes in different package.

In this example, we will create two packages p1 and p2. Class A in p1 is made public, to access it in p2. The method display in class A is protected and class B is inherited from class A and this protected method is then accessed by creating an object of class B.

```
//Java program to illustrate

//protected modifier

package p1;


//Class A

public class A

{

  protected void display()

   {

      System.out.println("GeeksforGeeks");

   }

}


//Java program to illustrate

//protected modifier

package p2;

import p1.*; //importing all classes in package p1


//Class B is subclass of A
```

```
class B extends A
{
  public static void main(String args[])
  {
     B obj = new B();
     obj.display();
  }

}
```

OUTPUT:

```
GeeksforGeeks.
```

QUESTION NUMBER 3: . a. What is inheritance and why it is used, discuss in detail ? b. Write a program using Inheritance class on Animal in java.

ANSWER:

A-

INHERITANCE:

A Programming technique that is used to reuse an existing class to build a new class is known as inheritance.
The new class inherits all the behavior of the original class.
The existing class that is used to create a new class is known as super class, base class or present class.
he new class that inherits the properties and functions of an existing class is known as subclass, derived class or child class. The inheritance relationship between the classes of a program is called a class hierarchy.
Inheritance is one of the most powerful features of object-oriented programming. The basic principle of inheritance is that each subclass shares common properties with the class from which it is derived. The child class inherits all the parent class and can add its own capabilities.

EXAMPLE:

Suppose we have a class named vehicle. The subclass of this class may share similar properties such as wheels, and motor etc. Additionally, a subclass may have its own particular characteristics. For example: a subclass may have seats for people but another subclass Truck may have space to carry goods.
A class of Animals can be divided into mammals, Amphibians, insects, reptiles.
A class of vehicles can be divided into cars, trucks, buses, and motorcycles. A class of shapes can be divided into subclasses lines, ellipses boxes.

USES OF INHERITANCE

The main use of inheritance is that it is

REUSABLE
SAVES TIME AND EFFORT

INCREASES PROGRAM STRUCTURE AND RELIABILITY

REUSABLE:
Inheritance allows developer to reuse existing code in many situations. A class can be created once and it can be reused again and again to create many subclasses.

SAVES TIME AND EFFORT:
Inheritance saves a lot of time and effort to write the same classes again. The reusability of existing classes allows the program to work only on new classes.

INCREASES PROGRAM STRUCTURE AND RELIABILITY
A super class is already compiled and tested properly. This class can be used in a new application without compiling it again. The use of existing class increases program reliability.

B-

```
class Animal{

void eat(){System.out.println("eating...");}

}

class Dog extends Animal{

void bark(){System.out.println("barking...");}

}
```

```
class Cat extends Animal{

void meow(){System.out.println("meowing...");}

}

class TestInheritance3{

public static void main(String args[]){

Cat c=new Cat();

c.meow();

c.eat();

//c.bark();//C.T.Error

}}
```

OUTPUT:

```
meowing...

eating...
```

QUESTION NUMBER 4: a. What is polymorphism and why it is used, discuss in detail ?

b. Write a program using polymorphism in a class on Employee in java.

ANSWER:

A-

POLYMORPHISM:

The word polymorphism is a combination of two words poly and morphism. Poly means many and morphism means form.

In object oriented programming, polymorphism is the ability of objects of different types to respond to functions of the same name. The user does not have to know the exact type of the object in advance. The behavior of the object can be implemented at run time. It is called late binding or dynamic binding. Polymorphism is implemented by using virtual functions.

USES OF POLYMORPHISM:

- The main use is the concept of extensively used in implementing inheritance.
- It plays an important role in allowing objects having different internal structure to share the same external interface.
- It is stated clear by itself, a one which is mapped for many.

B-

```
/* File name : Employee.java */


public class Employee {

private String name;

private String address;


private int number;
```

```
public Employee(String name, String address, int number) {
System.out.println("Constructing an Employee");

this.name = name;


this.address = address;

this.number = number;

}


public void mailCheck() {


System.out.println("Mailing a check to " + this.name + " " +
this.address);


}


public String toString() {


return name + " " + address + " " + number;

}
```

```
public String getName() {

return name;
}

public String getAddress() {
return address;
}

public void setAddress(String newAddress) { address = newAddress;
}

public int getNumber() {
return number;
}

}
```

Now suppose we extend Employee class as follow

```
/* File name : Salary.java */
```

```
}

public class Salary extends Employee { private double salary; // Annual
salary

public Salary(String name, String address, int number, double salary) {
super(name, address, number);
setSalary(salary);

}

public void mailCheck() {

System.out.println("Within mailCheck of Salary class ");
System.out.println("Mailing check to " + getName()
+       " with salary " + salary);
}

public double getSalary() {
return salary;

}
```

```
public void setSalary(double newSalary) { if(newSalary >= 0.0) {

salary = newSalary;

}

}


public double computePay() {


System.out.println("Computing salary pay for " + getName());


return salary/52;

}

}
```

Now, you study the following program carefully and try to determine its output

```
/* File name : VirtualDemo.java */ public class VirtualDemo {


public static void main(String [] args) {
```

```
Salary s = new Salary("Muhammad Saeed", "Peshawar, UP", 3,
3600.00);

Employee e = new Salary("James bond", "Europe, MA", 2, 2400.00);

System.out.println("Call mailCheck using Salary reference --
");

s.mailCheck();

System.out.println("\n Call mailCheck using Employee reference--");
e.mailCheck();

}
}
```

OUTPUT:

```
Constructing an Employee
Constructing an Employee
```

```
Call mailCheck using Salary reference -- Within mailCheck of Salary class

Mailing check to Muhammad Saeed with salary 3600.0


Call mailCheck using Employee reference--

Within mailCheck of Salary class

Mailing check to James bond with salary 2400.0
```

QUESTION NUMBER 5: a. Why abstraction is used in OOP, discuss in detail ?

b. Write a program on abstraction in java.

ANSWER:

ABSTRACTION IN OOP:

Abstraction is selecting data from a larger pool to show only the relevant details of the object to the user. Abstraction “shows” only the essential attributes and “hides” unnecessary information. It helps to reduce programming complexity and effort. It is one of the most important concepts of OOPs.

LETS STUDY ABSTRACTION CONCEPT WITH AN EXAMPLE:

Suppose you want to create a banking application and you are asked to collect all the information about your customer. There are chances that you will come up with following information about the customer

- Full name
- Address
- Contact Number
- Tax Information
- Favorite food
- Favorite actor
- Favorite Band
- Favorite Movie

But, not all of the above information is required to create a banking application.

So, you need to select only the useful information for your banking application from that pool. Data like name, address, tax information, etc. make sense for a banking application

- Full name
- Address
- Contact Number
- Tax Information

Since we have fetched/removed/selected the customer information from a larger pool, the process is referred as Abstraction.

However, the same information once extracted can be used for a wide range of applications. For instance, you can use the same data for hospital application, job portal application, a Government database, etc. with little or no modification. Hence, it becomes your Master Data. This is an advantage of Abstraction.

WHAT IS ABSTRACTION IN JAVA:

Abstraction in JAVA “shows” only the essential attributes and “hides” unnecessary details of the object from the user. In Java, abstraction is accomplished using Abstract classes, Abstract methods, and interfaces. Abstraction helps in reducing programming complexity and effort.

ABSTRACTION CLASS

A class which is declared “abstract” is called as an abstract class. It can have abstract methods as well as concrete methods. A normal class cannot have abstract methods.

ABSTRACTION METHOD

A method without a body is known as an Abstract Method. It must be declared in an abstract class. The abstract method will never be final because the abstract class must implement all the abstract methods.

ADVANTAGES OF ABSTRACTION:

- The main benefit of using an abstract class is that it allows you to group several related classes as siblings.
- Abstraction helps to reduce the complexity of the design and implementation process of software.

B-

```
// Abstract class
abstract class Animal {
  // Abstract method (does not have a body)
  public abstract void animalSound();
  // Regular method
  public void sleep() {
    System.out.println("Zzz");
  }
}
```

```
// Subclass (inherit from Animal)

class Cat extends Animal {

  public void animalSound() {

    // The body of animalSound() is provided here

    System.out.println("The cat says: meow meow");

  }

}


class MyMainClass {

  public static void main(String[] args) {

    Cat myCat = new Cat(); // Create a Cat object

    myCat.animalSound();

    myCat.sleep();

  }

}
```

OUTPUT:

```
The cat says: meow meow

Zzz
```