# Java Assessment 6
# Aaditya Kumar Muktavarapu
# HU21CSEN0100580

6. Create a class telephone with lift () and disconnected () methods as abstract methods create another class smart telephone and demonstrate polymorphism

```
import java.util.Scanner;

abstract class Telephone
{
   public abstract void lift();



   public abstract void disconnected();

}
class SmartTelephone extends Telephone
{
   public void lift()
   {
       System.out.println("Called lifted");
   }
   public void disconnected()
   {
       System.out.println("Call disconnected");
   }
}
public class phone {



   public static void main(String[] args)
   {
       System.out.println("Enter 1 to lift and 2 to disconnect");
       SmartTelephone phone = new SmartTelephone();
       Scanner sc = new Scanner(System.in);
       int x= sc.nextInt();

       if(x==1)
```

```
        {
            phone.lift();
        }
        else if(x==2)
        {
            phone.disconnected();
        }
        sc.close();
    }
}
```

Output

Enter 1 to lift and 2 to disconnect
2
Call disconnected

7. Design a vehicle class hierarchy in Java, and develop a program to demonstrate Polymorphism.

```
import java.util.Scanner;

abstract class Vehicle
{
    abstract void accelerate();

    abstract void brake();

    abstract void wheels();
}

class Car extends Vehicle
{
    void accelerate()
    {
        System.out.println("Drive slowly");
    }

    void brake()
    {
        System.out.println("Be careful");
```

```
    }

    void wheels()
    {
        System.out.println("Car has 4 wheels");
    }
}

class Bike extends Vehicle
{
    void accelerate()
    {
        System.out.println("Drive slowly");
    }

    void brake()
    {
        System.out.println("Be careful");
    }

    void wheels()
    {
        System.out.println("Bike has 2 wheels");
    }
}
class Auto extends Vehicle
{
    void accelerate()
    {
        System.out.println("Drive slowly");
    }

    void brake()
    {
        System.out.println("Be careful");
    }

    void wheels()
    {
        System.out.println("Auto has 3 wheels");
    }
}
```

```java
public class vehicles {
   public static void main(String[] args)
   {
       Car a=  new Car();
       Bike b= new Bike();
       Auto c= new Auto();
       Scanner sc=new Scanner(System.in);
       System.out.println("Enter 1 for car 2 for bike and 3 for auto.");
       int x=sc.nextInt();
       if(x==1)
       {
           a.accelerate();
           a.brake();
           a.wheels();
       }

       else if(x==2)
       {
           b.accelerate();
           b.brake();
           b.wheels();
       }

       else if(x==3)
       {
           c.accelerate();
           c.brake();
           c.wheels();
       }

       sc.close();
   }

}
```

Output

Enter 1 for car 2 for bike and 3 for auto.
2
Drive slowly
Be careful

Bike has 2 wheels

8. Write a program to find the roots of a quadratic equation using interface and packages.
• Declare an interface in package Quad1
• Declare another package Quad2 and implement the interface

```
import java.util.Scanner;

interface Quad1
{
   void calc();
}
class Quad2 implements Quad1
{
   public void calc()
   {
       Scanner sc = new Scanner(System.in);
       System.out.print("Enter a: ");
       int a=sc.nextInt();
       System.out.print("Enter b: ");
       int b=sc.nextInt();
       System.out.print("Enter c: ");
       int c=sc.nextInt();
       double disc=(b*b)-4*a*c;
       double x1=-b+Math.sqrt(disc)/(2*a);
       double x2=-b-Math.sqrt(disc)/(2*a);
       if (disc>0)
       {
           System.out.println("The roots are: "+x1+" and "+x2);
       }
       else if(disc==0)
       {
           double x= -b / (2 * a);
           System.out.println("The roots are: "+x);
       }
       else
       {
           System.out.println("The equation does not have any real roots");
       }
       sc.close();
```

```
    }
}
public class quadeq {
    public static void main(String[] args)
    {
        Quad2 ex = new Quad2();
        ex.calc();
    }

}
```

Output
Enter a: 1
Enter b: -7
Enter c: 12
The roots are: 7.5 and 6.5

9. Write a Program to generate Fibonacci Series by using Constructor to initialize the Data Members.

```
import java.util.Scanner;

class fib
{
    int a=0;
    int b=1;
    int c;

    fib(int a,int b)
    {
        this.a=a;
        this.b=b;
    }

    void display(int x)
    {
        System.out.print(a+" "+b);
        for(int i=2;i<x;i++)
        {
            c=a+b;
```

```
            System.out.print(" "+c);
            a=b;
            b=c;
        }
    }
}
public class fibonacci {
    public static void main(String[] args)
    {
        Scanner sc= new Scanner(System.in);
        System.out.println("Enter number of terms");
        int num=sc.nextInt();
        fib obj=new fib(0,1);
        obj.display(num);
        sc.close();
    }

}
```

Output

Enter number of terms

24

0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946 17711 28657%

10. Develop a program to demonstrate multiple inheritance through interface.

```
interface work
{
    void engineer();
}

interface hobby
{
    void musician();
}

class person implements work,hobby
{
    public void engineer()
    {
```

```
        System.out.println("Person is an engineer");
    }

    public void musician()
    {
        System.out.println("Person is a musician in free time");
    }
}
public class interinher {
    public static void main(String[] args)
    {
        person aadi = new person();
        aadi.engineer();
        aadi.musician();
    }

}
```

Output

Person is an engineer

Person is a musician in free time