NAME : UDHAYARAJAN J.
CLASS : MCA - I
DATE : 07/10/2025
SUB : JAVA PROGRAMMING
REG NO : 2202561
CCE : 1 (Unit I - open Book).

Q1) write Java code for Sum of digit (use scanner class eg I/P 1234 and 10).

sol:

```
import Java.util.scanner;
public class SumofDigit {
  public static void main (String[] args) {
    Scanner scanner = new Scanner(System.in);
    system.out.Print("Enter a number");
     int number = scanner.nextInt();

     int sum = 0;
     int temp = number;
     while ( temp != 0) {
        int digit = temp % 10;
        sum += digit;
        temp /= 10.
     }.
     System.out.println(" sum of digit " + number + " is " +
                             sum);
     Scanner.close();
   }.
}.
```

output :

```
Enter a number : 1234
Sum of digit of 1234 is : 10.
```

Q2). ~~write~~ what is a class in java ? create class Employee calculate salary. of employee and display name. and salary :

class :

In Java a class serves as a blueprint or template for creating objects. It's a fundamental concept in object oriented programming and define the structure. and behaviour that object of that class will posses.

ex:

A car class might have variable like. color and speed. and methode. like drive(). or break().

sol:

~~Class~~

```
import Java. util. Scanner;
class Employee {
        String name;
        double basicSalary;
        double bonus;

        double. calculateSalary() {
              return basicSalary + bonus;
        }.

        void. displayDetails() {
              double totalSalary = calculateSalary();
              System.out.println("Name :" + name);
              System. out. println(" total salay" + totalSalary);
        }.
}.
```

```
public class EmployeeMain {
    public static void main ( String[] args) {
        Scanner scanner = new Scanner (System.in);
        Employee emp = new Employee();
        System.out.print ("Emp Name :");
        emp.name = scanner.nextLine();
        System.out.print ("Emp Salary:");
        emp.basicSalary = scanner.nextDouble();
        System.out.print (" Emp bonus");
        emp.bonus = scanner.nextDouble();
        System.out.print("\n---- Details -----");
        emp.displayDetails();
        scanner.close();
    }
}
```

output:

```
Emp Name : Udhayarajan J.
Emp salary : 10000
Emp bonus : 5000.

----- Details -------
Name : Udhayarajan J
total salary : 10500.
```

Q3) what is Constructor? write overload constructor code for Item class.

Constructor:

A Constructor in Java is a special nethod. that is automatically when an object is created to is used to initialize the object's variable.

- The construtor name must be the same as the class Name.
- it does not have return type (not even void)
- it run automatically when you create an object using (new).
- ~~you~~ can have multiple Constructor with different parameter list - this is called constructor overloading.

sol.

```
class Item {
    int itemID;
    String itemName;
    double price;

    Item() {          // Defaut constructor.
        itemID = 0;
        itemName = "unknown";
        price = 0.0;
    }

    Item(int ID, String name) { // paramotarized Constructor
        itemID = ID;                   (2 param).
        itemName = name;
        price = 0.0;
    }

    Item(int ID, String name, double price) {
        itemID = ID;            // paramotarized Constructor.
        ItemName = name;           (3 Param)
        price = price;
    }
```

```
        void display() {
            system.out.println(" ID : " + item ID);
            System.out.println("Name : " + itemName);
            System.out.println("price : " + price);
        }
    }
    public class ItemMain {
        public static void main (String [] args) {
            Item item1 = new Item();
            Item item2 = new Item(101, "Note book");
            Item item3 = new Item(102, "pen", 10.5);

            item1.display();
            item2.display();
            item3.display();
        }
    }
```

output :

```
ID : 0
Name : unknown
price : 0.0.

ID : 101
Name : Note book
price : 0.0.

ID : 102.
Name : pen.
price : 10.5.
```

Q4) Implement Multilevel inheritance to inherit feature of Animal, cat and dog?

Multilevel inheritance:

It means a class derived from another derived class, forming a chain of inheritance.

Animal → cat → Dog.

- cat inherits from Animal.
- Dog inherits from cat.
- Thus, Dog indirectly inherit from Animal.

Sol:

```
class Animal {
        void eat() {
            System.out.println("Animal eats food");
        }
}

class Cat extends Animal {
        void meow() {
            System.out.println("Cat meow");
        }
}

class Dog extends Cat {
        void bark() {
            System.out.println("Dog barks");
        }
}
```

```
public class Multilevelinheritance {
    public static void main(String [] args) {
        Dog dog = new Dog();

        dog.eat();
        dog.meow();
        dog.bark();
    }
}
```

output:

```
Animal eat food.
cat meow.
Dog barks
```

Q5). what is polymorphism? write Java code to calculate area of different shapes by using method. overloading.

polymorphism:

- poly morphism in Java is a core concept of object oriented. programming (oop). that allow to. objects to take on "many forms"
- The form polymorphism is derived from Greek words: poly means many; morphs means forms.
- it allow one name tobe used for. different behaviour.
- in Java. polymorphism let the same method or operator act differently depending on the. object or data passed.

Types of polymorphism:

- Compile time polymorphism (method overloading)
- Runtime polymorphism (method overriding)

Method overloading:

- Achieve when multiple methods in the same class have the same name but different parameter.
- Decided by the compiler.
-

method overriding:

- Acheive when a subclass provides a specific implementation of a method already defined inits parent class.
- Decided by the JVM at runtime.

sol:

```
class shape {
    double area(double side) {
        return side * side;
    }

    double area(double length, double width) {
        return length * width;
    }

    double area(float radius) {
        return 3.14159 * radius * radius;
    }
}
```

```
public class polymorphismExample {
    public static void main(String[] args) {
        shape Shape = new Shape();

         system.out.println("Area of Square : "+ Shape.
                                              area(5.0));
        system.out.println("Area of rectangle: "+ shape.area
                                              (5.0, 10.0));
        system.out.println("Area of circle: "+ shape.area
                                              (3.5f));
    }
}
```

output:

```
Area of Square : 25.0
Area of rectangle : 50.0
Area of circle : 38.4846975
```