Class 10

Chapter – 8

Iterative Construct in Java

A. TICK (✓) THE CORRECT ANSWER

1. Output of the code

```
int i, j;
for(i = 2, j = 4; i <= 4; i = i + 1)
{
    if(j % i == 0)
        break;
}
System.out.println(i);
```

- Answer: **a. 2**

Explanation:
Start: i = 2, j = 4.
First iteration: condition (i <= 4) is true.
Inside loop: j % i = 4 % 2 = 0 → condition of if is true → `break;` executes.
Loop stops with i still equal to 2. `println(i)` prints 2.

2. Which is not a loop in Java?

- Answer: **d. switch**

Explanation: `for`, `while` and `do-while` are loop (iterative) statements; `switch` is a selection statement.

3. Which is a null loop?

- Answer: **c. Both a and b**

Explanation:
`while(i < 5) {}` has an empty body.
`for(i = 0; i <= 10; i++);` has a semicolon as the body (empty statement).
Both are null (bodyless) loops.

4. Code:

```
int a = 5, b = 2;
if(a > b)
{
    a = a + 1;
    b = b * 2;
}
System.out.print(a + ":" + b);
```

- Answer: **c. 6:4**

Explanation: `a > b` (5 > 2) is true, so body executes.
`a = 5 + 1 = 6`, `b = 2 * 2 = 4`.
Printed text is `6:4`.

5. How many times does this loop execute?

```
int i = 2, j = 10;
while(i < j)
{
    i = i / j;
    System.out.println(i);
}
```

- Answer: **d. infinite**

Explanation:
Start: i = 2.

1st time: 2 < 10 → true, `i = 2 / 10 = 0` (integer division), prints 0.
Now i = 0.
Again: 0 < 10 → true; `i = 0 / 10 = 0`.
i remains 0 forever; condition is always true → infinite loop.

B. FILL IN THE BLANKS

1. To find all the even numbers from 2 to 20, the loop will execute for **10** times.
   (2, 4, 6, 8, 10, 12, 14, 16, 18, 20 → 10 numbers)
2. In for loop, `i <= 10` is known as **test condition** (or condition).
3. `i++` in while loop is known as **increment statement** (update statement).
4. To print the sum of 10 natural numbers using loop, `System.out.print(sum)` statements should be written **outside** the loop.
5. If `i = 1`, then the loop `while(i < 5) { i++; }` will execute **4** times.
   (i values: 1→2, 2→3, 3→4, 4→5, then 5 < 5 is false.)

C. SHORT ANSWER TYPE QUESTIONS

C1. SERIES – DISPLAY ONLY

1(i). 1 2 3 4 5 … 10

```
class Series1
{
    public static void main(String[] args)
    {
        for(int i = 1; i <= 10; i++)
            System.out.print(i + " ");
    }
}
```

1(ii). 1 3 5 7 9 … 19 (odd numbers)

```
class Series2
{
    public static void main(String[] args)
    {
        for(int i = 1; i <= 19; i = i + 2)
            System.out.print(i + " ");
    }
}
```

1(iii). 0 3 6 9 … 99 (multiples of 3 starting from 0)

```
class Series3
{
    public static void main(String[] args)
    {
        for(int i = 0; i <= 99; i = i + 3)
            System.out.print(i + " ");
    }
}
```

1(iv). 5 10 15 20 … 50

```
class Series4
{
    public static void main(String[] args)
    {
        for(int i = 5; i <= 50; i = i + 5)
            System.out.print(i + " ");
    }
}
```

1(v). 1/3 2/6 3/9 … 10/30

```
class Series5
{
    public static void main(String[] args)
    {
        for(int n = 1; n <= 10; n++)
        {
            int num = n;
            int den = 3 * n;
            System.out.print(num + "/" + den + "  ");
        }
    }
}
```

1(vi). 2/4 4/6 6/8 … 10/12

```
class Series6
{
    public static void main(String[] args)
    {
        for(int num = 2; num <= 10; num = num + 2)
        {
            int den = num + 2;
            System.out.print(num + "/" + den + "  ");
        }
    }
}
```

C2. SERIES – CALCULATE VALUE

2(i). s1 = 1 + 2 + 3 + … + 20

```
class Sum1To20
{
    public static void main(String[] args)
    {
        int sum = 0;
        for(int i = 1; i <= 20; i++)
            sum = sum + i;

        System.out.println("Sum = " + sum);
    }
}
```

2(ii). p1 = 1 * 3 * 5 * 7 * 9

```
class ProductOdd
{
    public static void main(String[] args)
    {
        int prod = 1;
        for(int i = 1; i <= 9; i = i + 2)
            prod = prod * i;

        System.out.println("Product = " + prod);
    }
}
```

2(iii). s2 = 1 + 11 + 111 + 1111 + 11111

```
class Series111
{
    public static void main(String[] args)
    {
        int term = 0;
        int sum = 0;
```

```
        for(int i = 1; i <= 5; i++)
        {
            term = term * 10 + 1;   // 1, 11, 111, ...
            sum = sum + term;
        }

        System.out.println("Sum = " + sum);
    }
}
```

2(iv). p2 = 2 * 4 * 8 * 16 * 32

```
class ProductP2
{
    public static void main(String[] args)
    {
        int term = 2;
        int prod = 1;

        for(int i = 1; i <= 5; i++)
        {
            prod = prod * term;
            term = term * 2;     // next term doubled
        }

        System.out.println("Product = " + prod);
    }
}
```

2(v). s3 = 1! + (12) + (123) + (1234) + (12345)

```
class SumFactorialLike
{
    public static void main(String[] args)
    {
        int sum = 0;
        int term = 1;
        for(int i = 1; i <= 5; i++)
        {
            term = term * i;   // running product 1, 1*2, 1*2*3, ...
            sum = sum + term;
        }
        System.out.println("Sum = " + sum);
    }
}
```

2(vi). s4 = (1)/(1) + (1+2)/(12) + (1+2+3)/(123) + ... + (1+2+...+10)/(12*…*10)

```
class SumS4
{
    public static void main(String[] args)
    {
        double sum = 0.0;
        int numSum = 0;      // numerator: 1, 1+2, 1+2+3, ...
        long fact = 1;       // denominator: factorial

        for(int i = 1; i <= 10; i++)
        {
            numSum = numSum + i;
            fact = fact * i;
            sum = sum + (double)numSum / fact;
        }

        System.out.println("Sum = " + sum);
    }
}
```

2(vii) p=1/2*2/3*3/4*…….9/10

C3. LCM OF TWO NUMBERS

```
import java.util.Scanner;

class LCMOfTwo
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter two numbers: ");
        int a = sc.nextInt();
        int b = sc.nextInt();

        int max = (a > b) ? a : b;
        int lcm = max;

        while(true)
        {
            if(lcm % a == 0 && lcm % b == 0)
                break;
            lcm++;
        }

        System.out.println("LCM = " + lcm);
    }
}
```

C4. DIGITS IN REVERSE WITH COMMAS

Example: n = 3456 → 6,5,4,3

```
import java.util.Scanner;

class DigitsReverseComma
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();

        boolean first = true;
        while(n > 0)
        {
            int d = n % 10;
            if(first)
            {
                System.out.print(d);
                first = false;
            }
            else
            {
```

```
                System.out.print("," + d);
            }
            n = n / 10;
        }
    }
}
```

C5. SUM OF EVEN DIGITS AND PRODUCT OF ODD DIGITS

```
import java.util.Scanner;

class SumEvenProductOdd
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();

        int sumEven = 0;
        int prodOdd = 1;
        boolean oddFound = false;

        while(n > 0)
        {
            int d = n % 10;
            if(d % 2 == 0)
                sumEven = sumEven + d;
            else
            {
                prodOdd = prodOdd * d;
                oddFound = true;
            }
            n = n / 10;
        }

        System.out.println("Sum of even digits is: " + sumEven);
        if(oddFound)
            System.out.println("Product of odd digits is: " + prodOdd);
        else
            System.out.println("No odd digits.");
    }
}
```

C6. GENERAL SERIES (VALUES FROM USER)

```
import java.util.Scanner;

class GeneralSeries
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);

        // i) a + a^2 + a^3 + ... + a^n
        System.out.print("Enter a and n for (a + a^2 + ... + a^n): ");
        int a = sc.nextInt();
        int n = sc.nextInt();
        double sum1 = 0;
        for(int i = 1; i <= n; i++)
            sum1 = sum1 + Math.pow(a, i);
```

```
        System.out.println("Sum1 = " + sum1);

        // ii) s = x^2 + 2x^2 + 3x^2 + ... + n x^2
        System.out.print("Enter x and n for (x^2 + 2x^2 + ... + n x^2): ");
        int x = sc.nextInt();
        n = sc.nextInt();
        double sum2 = 0;
        for(int i = 1; i <= n; i++)
            sum2 = sum2 + i * x * x;
        System.out.println("Sum2 = " + sum2);

        // iii) p1 = 1^1 * 2^2 * 3^3 * ... * n^n
        System.out.print("Enter n for product 1^1 * 2^2 * ... * n^n: ");
        n = sc.nextInt();
        double prod1 = 1;
        for(int i = 1; i <= n; i++)
            prod1 = prod1 * Math.pow(i, i);
        System.out.println("Product1 = " + prod1);

        // iv) s = 1/sqrt(2) + 1/sqrt(4) + ... + 1/sqrt(10)
        double sum3 = 0;
        for(int d = 2; d <= 10; d = d + 2)
            sum3 = sum3 + 1.0 / Math.sqrt(d);
        System.out.println("Sum3 = " + sum3);

        // v) s = 1/12 + 1/14 + 1/16 + ... + 1/110
        double sum4 = 0;
        for(int d = 12; d <= 110; d = d + 2)
            sum4 = sum4 + 1.0 / d;
        System.out.println("Sum4 = " + sum4);

        // vi) s = a^2 - a^3 + a^4 - a^5 + ... up to n terms
        System.out.print("Enter a and n for (a^2 - a^3 + a^4 - ...): ");
        a = sc.nextInt();
        n = sc.nextInt();
        double sum5 = 0;
        int power = 2;
        int sign = 1;   // + then -
        for(int i = 1; i <= n; i++)
        {
            sum5 = sum5 + sign * Math.pow(a, power);
            power++;
            sign = -sign;
        }
        System.out.println("Sum5 = " + sum5);

        // vii) p = (x+1)*(x+3)*(x+5)*...*(x+n)
        System.out.print("Enter x and n for product (x+1)*(x+3)*...*(x+n): ");
        x = sc.nextInt();
        n = sc.nextInt();
        double prod2 = 1;
        for(int k = 1; k <= n; k = k + 2)
            prod2 = prod2 * (x + k);
        System.out.println("Product2 = " + prod2);
    }
}
```

C7. NIVEN NUMBER (HARSHAD NUMBER)

```
import java.util.Scanner;
```

```
class NivenNumber
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();

        int temp = n;
        int sumDigits = 0;
        while(temp > 0)
        {
            sumDigits = sumDigits + temp % 10;
            temp = temp / 10;
        }

        if(n % sumDigits == 0)
            System.out.println(n + " is a Niven number.");
        else
            System.out.println(n + " is not a Niven number.");
    }
}
```

C8. AUTOMORPHIC NUMBER

Definition: A number is Automorphic if its square ends with the same digits as the number itself.

```
import java.util.Scanner;

class AutomorphicNumber
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();

        int square = n * n;
        int temp = n;
        boolean auto = true;

        while(temp > 0)
        {
            if(temp % 10 != square % 10)
            {
                auto = false;
                break;
            }
            temp = temp / 10;
            square = square / 10;
        }

        if(auto)
            System.out.println("Automorphic number");
        else
            System.out.println("Not Automorphic");
    }
}
```

C9. KRISHNAMURTHY NUMBER

Definition: Sum of factorial of its digits equals the number.

```
import java.util.Scanner;

class KrishnamurthyNumber
{
    static int fact(int n)
    {
        int f = 1;
        for(int i = 1; i <= n; i++)
            f = f * i;
        return f;
    }

    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();

        int temp = n;
        int sum = 0;
        while(temp > 0)
        {
            int d = temp % 10;
            sum = sum + fact(d);
            temp = temp / 10;
        }

        if(sum == n)
            System.out.println("Krishnamurthy number");
        else
            System.out.println("Not Krishnamurthy");
    }
}
```

C10. SPY NUMBER

Definition: Sum of digits equals product of digits.

```
import java.util.Scanner;

class SpyNumber
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();

        int temp = n;
        int sum = 0;
        int prod = 1;

        while(temp > 0)
        {
            int d = temp % 10;
            sum = sum + d;
            prod = prod * d;
            temp = temp / 10;
        }

        if(sum == prod)
            System.out.println("Spy number");
```

```
        else
            System.out.println("Not a spy number");
    }
}
```

C11. MENU – PERFECT OR TRIMORPHIC

```
import java.util.Scanner;

class PerfectTrimorphicMenu
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("1. Check Perfect Number");
        System.out.println("2. Check Trimorphic Number");
        System.out.print("Enter your choice: ");
        int ch = sc.nextInt();

        System.out.print("Enter number: ");
        int n = sc.nextInt();

        switch(ch)
        {
            case 1:
                int sum = 0;
                for(int i = 1; i <= n / 2; i++)
                    if(n % i == 0)
                        sum = sum + i;
                if(sum == n)
                    System.out.println("Perfect number");
                else
                    System.out.println("Not perfect");
                break;

            case 2:
                long cube = (long)n * n * n;
                int temp = n;
                boolean tri = true;
                while(temp > 0)
                {
                    if(temp % 10 != cube % 10)
                    {
                        tri = false;
                        break;
                    }
                    temp = temp / 10;
                    cube = cube / 10;
                }
                if(tri)
                    System.out.println("Trimorphic number");
                else
                    System.out.println("Not trimorphic");
                break;

            default:
                System.out.println("Wrong choice");
        }
    }
}
```

C12. MENU – PRINT SERIES (SWITCH)

i) s = 1 + 12 + 123 + 1234 + 12345

ii) p = 1/1 * 2/4 * 3/9 * … up to n terms

```
import java.util.Scanner;

class SeriesMenu
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("1. s = 1 + 12 + 123 + 1234 + 12345");
        System.out.println("2. p = 1/1 * 2/4 * 3/9 * ... up to n terms");
        System.out.print("Enter your choice: ");
        int ch = sc.nextInt();

        switch(ch)
        {
            case 1:
                int term = 0;
                int sum = 0;
                for(int i = 1; i <= 5; i++)
                {
                    term = term * 10 + i;   // 1, 12, 123, 1234, 12345
                    sum = sum + term;
                }
                System.out.println("Sum = " + sum);
                break;

            case 2:
                System.out.print("Enter n: ");
                int n = sc.nextInt();
                double prod = 1.0;
                for(int i = 1; i <= n; i++)
                {
                    double num = i;
                    double den = i * i;   // 1,4,9,...
                    prod = prod * (num / den);
                }
                System.out.println("Product = " + prod);
                break;

            default:
                System.out.println("Wrong choice");
        }
    }
}
```

C13. MENU – TABLE OR FRACTION SERIES

i) Print first n natural numbers with their squares and cubes.

ii) s = 1/1^3 - 1/2^3 + 1/3^3 - … up to n terms.

```
import java.util.Scanner;

class TableOrSeries
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("1. Table of n numbers with square and cube");
        System.out.println("2. s = 1/1^3 - 1/2^3 + 1/3^3 - ... up to n terms");
```

```
        System.out.print("Enter your choice: ");
        int ch = sc.nextInt();

        System.out.print("Enter n: ");
        int n = sc.nextInt();

        switch(ch)
        {
            case 1:
                System.out.println("No\tSquare\tCube");
                for(int i = 1; i <= n; i++)
                    System.out.println(i + "\t" + (i*i) + "\t" + (i*i*i));
                break;

            case 2:
                double sum = 0.0;
                int sign = 1;
                for(int i = 1; i <= n; i++)
                {
                    sum = sum + sign * 1.0 / (i*i*i);
                    sign = -sign;
                }
                System.out.println("Sum = " + sum);
                break;

            default:
                System.out.println("Wrong choice");
        }
    }
}
```

C14. SUM OF SERIES WITH FACTORIALS

Series: s = x + x^2 / 2! + x^3 / 3! + … + x^n / n!

```
import java.util.Scanner;

class SeriesFactorial
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter x: ");
        double x = sc.nextDouble();
        System.out.print("Enter n: ");
        int n = sc.nextInt();

        double sum = 0.0;
        double fact = 1.0;

        for(int i = 1; i <= n; i++)
        {
            fact = fact * i;                    // i!
            sum = sum + Math.pow(x, i) / fact;
        }

        System.out.println("Sum = " + sum);
    }
}
```

C15. TAX FOR 5 EMPLOYEES

Tax slabs:
Up to 20000 → 0%
Next 15000 → 5.5%
Next 30000 → 7.5%
Above 65000 → 10%
(Here tax is computed **slab-wise**.)

```
import java.util.Scanner;

class TaxForEmployees
{
    static double taxOnSalary(double sal)
    {
        double tax = 0.0;

        if(sal > 65000)
        {
            tax = tax + (sal - 65000) * 10.0 / 100.0;
            sal = 65000;
        }
        if(sal > 35000)
        {
            tax = tax + (sal - 35000) * 7.5 / 100.0;
            sal = 35000;
        }
        if(sal > 20000)
        {
            tax = tax + (sal - 20000) * 5.5 / 100.0;
            sal = 20000;
        }
        // Up to 20000 → no tax

        return tax;
    }

    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        double totalTax = 0.0;

        for(int i = 1; i <= 5; i++)
        {
            System.out.print("Enter salary of employee " + i + ": ");
            double sal = sc.nextDouble();
            double tax = taxOnSalary(sal);
            totalTax = totalTax + tax;
            System.out.println("Tax for employee " + i + " = " + tax);
        }

        System.out.println("Total tax collected = " + totalTax);
    }
}
```

D. Assertion and Reasoning based questions

The following questions consists of two statements Assertion (A) and Reason (R) answer these questions by selecting the appropriate option given below :

Options:

a. Both A and R are true, and R is the correct explanation of A
b. Both A and R are true, but R is not the correct explanation of A
c. A is true, but R is false
d. A is false, but R is true

Q1.

Assertion (A): The "break" statement can be used to exit from a loop prematurely.

Reason (R): The "break" statement terminates the loop and transfers control to the statement immediately following the loop.

Answer: a. Both A and R are true, and R is the correct explanation of A

Explanation:

- The **break statement** stops the loop immediately.
- Control moves to the **next statement after the loop**.
- This explains how it exits the loop prematurely.

Q2.

Assertion (A): The "continue" statement in a loop causes the loop to terminate.

Reason (R): The "continue" statement skips the remaining code in the loop's current iteration and proceeds with the next iteration.

Answer: d. A is false, but R is true

Explanation:

- Assertion is **false** because `continue` does NOT terminate the loop.
- It only **skips current iteration** and continues the loop.
- Reason is **true**, correctly describing the behavior of `continue`.

****