Class 9(2026-27)
Computer Applications
Chapter 9
Iterative Constructs in Java

A. Tick (✓) the Correct Answer

Question 1

What is the output of the following program snippet?

```
int i=1, s=0;

do
{
    if(i%4==0)
    {
        System.out.println((i*2)+",");
    }
    i++;
}
while(i<=10);
```

a. 4,8
b. 8,16
c. 1,2,3.....10
d. None of these

Answer:

✓ **b. 8,16**

Explanation:

Numbers divisible by 4 between 1 and 10 are:
`4 and 8`
Their doubles are:
`8 and 16`
Hence output:
`8,16`

Question 2

`"for(i=10;i>=1;i++)"` loop is a _____ loop.

a. exit
b. entry
c. infinite
d. finite

Answer:

✓ **c. infinite**

Explanation:

- `i++` increases the value of `i`
- Condition is `i>=1`
- Since `i` keeps increasing forever, the loop never ends.

Question 3

The following program segment prints:

```
int a=5,b=2;

do
{
    if(a>b)
    {
        a=a+1;
        b=b*2;
    }
}
while(a==0);

System.out.print(a + ":" + b);
```

a. 6:2
b. 6:3
c. 6:4
d. 6:1

Answer:

✓ **c. 6:4**

Explanation:

Initially:

```
a = 5
b = 2
```

Condition:

```
a>b → true
```

So:

```
a = 6
b = 4
```

Loop condition:

```
a==0 → false
```

Output:

```
6:4
```

Question 4

What will be the output of the following code?

```
int m = 2;
int n = 15;

for(int i = 1; i < 5; i++);

m++;
--n;

System.out.println("m = " + m);
System.out.println("n = " + n);
```

a. m=3, n=11
b. m=1, n=14
c. m=3, n=14
d. No output

Answer:

✓ **c. m=3, n=14**

Explanation:

- `for` loop has a semicolon `;`
- So it has an empty body.
- Then:

`m++ → 3`
`--n → 14`
Output:
`m = 3`
`n = 14`

B. Fill in the Blanks

Question 1

_____ loops have an empty loop body.

Answer:

✅ **Null**

Question 2

The two jump statements are _____ and _____.

Answer:

✅ **break** and **continue**

Question 3

To execute a loop 10 times,

`for(i=3; i<= _____ ; i=i+3)`

Answer:

✅ **30**

Explanation:

Series:
`3,6,9,12....30`
Total = 10 terms

Question 4

`for(i=10;i<10;i++)` loop executes for _____ times.

Answer:

✅ **0**

Explanation:

Condition:
`10 < 10`
is false initially.

Question 5

_____ loop checks the condition at the time of entry.

Answer:

✅ **while**

Question 6

Both _____ and do-while are suitable in situations where the number of iterations is not known.

Answer:

✓ **while**

C. Answer the Following Questions

Question 1

What are different parts of a loop?

Answer:

The different parts of a loop are:

1. Initialization
2. Condition/Test Expression
3. Increment/Decrement
4. Body of the loop

Example

```
for(i=1; i<=5; i++)
{
   System.out.println(i);
}
```

Part	Example
Initialization	`i=1`
Condition	`i<=5`
Increment	`i++`
Body	`System.out.println(i)`

Question 2

Define two categories of loops.

Answer:

1. Entry Controlled Loop

- Condition is checked before entering the loop.
- Example:

```
while
for
```

2. Exit Controlled Loop

- Condition is checked after executing the loop body.
- Example:

```
do-while
```

Question 3

Differentiate between Infinite loop and Finite loop.

Finite Loop	Infinite Loop
Ends after some iterations	Never ends
Condition becomes false	Condition always true

Finite Loop	Infinite Loop
Useful in normal programming	Causes endless execution

Example of Finite Loop

```
for(i=1;i<=5;i++)
```

Example of Infinite Loop

```
for(;;)
```

Question 4

What are delay loops? Explain with example.

Answer:

A delay loop is a loop used to create time delay in a program.

Example

```
for(int i=1;i<=10000;i++)
{
}
```

Explanation:

- Loop runs many times.
- Creates small delay in execution.

Question 5

What is interconversion of loops?

Answer:

Changing one type of loop into another type is called interconversion of loops.

Example:

- `for` loop into `while`
- `while` into `do-while`

Example

for loop

```
for(i=1;i<=5;i++)
{
   System.out.println(i);
}
```

Equivalent while loop

```
i=1;

while(i<=5)
{
   System.out.println(i);
   i++;
}
```

D. Assertion and Reasoning Based Question

Assertion (A):

A loop is a programming construct that does not repeat a set of statements until a certain condition is met.

Reason (R):

A loop can be repeated any number of times, depending on the user's requirements.

Answer:

✅ **d. A is false, but R is true**

Explanation:

- Assertion is false because loops DO repeat statements.
- Reason is true.

E. More Unsolved Programs

1(a). Series Program

Display:

```
1,3,5,7,9.....99
```

Answer

```
class Series1
{
    public static void main(String args[])
    {
        for(int i=1;i<=99;i=i+2)
        {
            System.out.print(i + " ");
        }
    }
}
```

1(b). Series Program

Display:

```
20,18,16.....2
```

Answer

```
class Series2
{
    public static void main(String args[])
    {
        for(int i=20;i>=2;i=i-2)
        {
            System.out.print(i + " ");
        }
    }
}
```

1(c). Series Program

Display:

```
2,4,8,16.....256
```

Answer

```
class Series3
{
    public static void main(String args[])
    {
        for(int i=2;i<=256;i=i*2)
        {
            System.out.print(i + " ");
        }
    }
}
```

1(d). Series Program

Display:

```
1/3 2/6 3/9.....10/30
```

Answer

```
class Series4
{
    public static void main(String args[])
    {
        for(int i=1;i<=10;i++)
        {
            System.out.print(i + "/" + (i*3) + " ");
        }
    }
}
```

1(e). Series Program

Display:

```
1,12,123,1234,12345
```

Answer

```
class Series5
{
    public static void main(String args[])
    {
        int n=0;

        for(int i=1;i<=5;i++)
        {
            n = n*10 + i;
            System.out.print(n + " ");
        }
    }
}
```

1(f). Series Program

Display:

```
1,11,111,1111,11111
```

Answer

```
class Series6
{
    public static void main(String args[])
    {
        int n=0;

        for(int i=1;i<=5;i++)
        {
            n = n*10 + 1;
            System.out.print(n + " ");
        }
    }
}
```

2(a). Sum Series Program

Find:

```
s = 1+4+9+16+25
```

Answer

```
class SumSeries1
{
    public static void main(String args[])
    {
        int s=0;

        for(int i=1;i<=5;i++)
        {
            s = s + (i*i);
        }

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

2(b). Product Series Program

Find:

`p = 10*9*8*7.....2*1`

Answer

```
class ProductSeries
{
    public static void main(String args[])
    {
        int p=1;

        for(int i=10;i>=1;i--)
        {
            p = p * i;
        }

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

2(c). Sum Series Program

Find:

`s = 1 + 11 + 111 + 1111 + 11111`

Answer

```
class SumSeries2
{
    public static void main(String args[])
    {
        int s=0, n=0;

        for(int i=1;i<=5;i++)
        {
            n = n*10 + 1;
            s = s + n;
        }

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

Explanation

Generated numbers:
`1,11,111,1111,11111`
All are added in variable `s`.

2(d). Sum Series Program

Find:

```
s = 10 + 20 + 30 + 40 + ... + 100
```

Answer

```
class SumSeries3
{
    public static void main(String args[])
    {
        int s=0;

        for(int i=10;i<=100;i=i+10)
        {
            s = s + i;
        }

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

Explanation

The loop adds multiples of 10 from 10 to 100.

2(e). Product Series Program

Find:

```
p = 1/2 * 2/3 * 3/4 * ... * 9/10
```

Answer

```
class ProductFraction
{
    public static void main(String args[])
    {
        double p=1.0;

        for(int i=1;i<=9;i++)
        {
            p = p * ((double)i/(i+1));
        }

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

Explanation

Fractions multiplied:

```
1/2 × 2/3 × 3/4 × ... × 9/10
```

2(f). Series Program

Find:

```
s = 2 + 5 + 10 + 17 + ...
```

Answer

```
import java.util.*;

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

```
    {
        Scanner sc = new Scanner(System.in);

        int n,s=0;

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

        for(int i=1;i<=n;i++)
        {
            s = s + (i*i + 1);
        }

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

Explanation

Pattern:

$1^2+1 = 2$
$2^2+1 = 5$
$3^2+1 = 10$
$4^2+1 = 17$

---

3. Neon Number Program

Answer

```
import java.util.*;

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

        int n,sq,sum=0,d;

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

        sq=n*n;

        while(sq>0)
        {
            d=sq%10;
            sum=sum+d;
            sq=sq/10;
        }

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

---

4. Digits Separated by Comma Program

Answer

```
import java.util.*;

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

```
        int n,d;

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

        while(n>0)
        {
            d=n%10;
            System.out.print(d + ",");
            n=n/10;
        }
    }
}
```

---

5. Reverse Number Program

Answer

```
import java.util.*;

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

        int n,rev=0,d,original,diff;

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

        original=n;

        while(n>0)
        {
            d=n%10;
            rev=rev*10+d;
            n=n/10;
        }

        diff=Math.abs(original-rev);

        System.out.println("Reverse = " + rev);
        System.out.println("Absolute Difference = " + diff);
    }
}
```

---

6. Disarium Number Program

Question

Write a program to check whether a number is a Disarium Number or not.
A number is Disarium if the sum of its digits powered with their respective positions is equal to the number itself.
Example:

$135 = 1^1 + 3^2 + 5^3 = 135$

Answer

```
import java.util.*;

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

```
        int n,temp,d,sum=0,count=0;

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

        temp=n;

        while(temp>0)
        {
            count++;
            temp=temp/10;
        }

        temp=n;

        while(temp>0)
        {
            d=temp%10;
            sum = sum + (int)Math.pow(d,count);
            count--;
            temp=temp/10;
        }

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

7(a). Series Program

Find:

$s = a^1 + a^2 + a^3 + ... + a^n$

Answer

```
import java.util.*;

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

        int a,n;
        double s=0;

        System.out.print("Enter value of a: ");
        a=sc.nextInt();

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

        for(int i=1;i<=n;i++)
        {
            s = s + Math.pow(a,i);
        }

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

7(b). Series Program

Find:

$s = x^2 + 2x^2 + 3x^2 + ... + nx^2$

Answer

```
import java.util.*;

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

        int x,n,s=0;

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

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

        for(int i=1;i<=n;i++)
        {
            s = s + (i*x*x);
        }

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

7(c). Product Series Program

Find:

$p = 1^1 * 2^2 * 3^3 * ... * n^n$

Answer

```
import java.util.*;

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

        int n;
        double p=1;

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

        for(int i=1;i<=n;i++)
        {
            p = p * Math.pow(i,i);
        }

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

7(d). Series Program

Find:

$s = 1 + 2 + 4 + 7 + 11 + 16 + ...$

Answer

```
import java.util.*;
```

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

        int n,term=1,diff=1,s=0;

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

        for(int i=1;i<=n;i++)
        {
            s=s+term;
            term=term+diff;
            diff++;
        }

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

Explanation

Difference increases by 1:
`+1,+2,+3,+4...`

7(e). Series Program

Find:

`s = 2/12 + 4/14 + 6/16 + ... + 1/10`

Answer

```
class FractionSeries
{
    public static void main(String args[])
    {
        double s=0;

        for(int i=2,j=12;i<=10;i=i+2,j=j+2)
        {
            s = s + ((double)i/j);
        }

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

7(f). Series Program

Find:

`a/2 + a/4 + a/6 + ... + a/10`

Answer

```
import java.util.*;

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

        int a;
        double s=0;
```

```
        System.out.print("Enter value of a: ");
        a=sc.nextInt();

        for(int i=2;i<=10;i=i+2)
        {
            s = s + ((double)a/i);
        }

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

7(g). Series Program

Find:

$s = a^1 - a^2 + a^3 - a^4 + ...$

Answer

```
import java.util.*;

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

        int a,n;
        double s=0;

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

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

        for(int i=1;i<=n;i++)
        {
            if(i%2==0)
                s = s - Math.pow(a,i);
            else
                s = s + Math.pow(a,i);
        }

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

7(h). Product Series Program

Find:

$p = (x+1)(x+3)(x+5)...(x+n)$

Answer

```
import java.util.*;

class ProductSeries2
{
    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in);

        int x,n;
        long p=1;

        System.out.print("Enter x: ");
```

```
        x=sc.nextInt();

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

        for(int i=1;i<=n;i=i+2)
        {
            p = p * (x+i);
        }

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

---

8. Niven Number Program

Answer

```
import java.util.*;

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

        int n,temp,d,sum=0;

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

        temp=n;

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

        if(n%sum==0)
            System.out.println("Niven Number");
        else
            System.out.println("Not Niven Number");
    }
}
```

---

9. Automorphic Number Program

Answer

```
import java.util.*;

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

        int n,sq;

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

        sq=n*n;

        if(sq%10==n || sq%100==n || sq%1000==n)
            System.out.println("Automorphic Number");
        else
```

```
            System.out.println("Not Automorphic Number");
    }
}
```

10(a). Menu Driven Series Program

Find:

```
s = 1 + 12 + 123 + 1234 + 12345
```

Answer

```
import java.util.*;

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

        int choice;

        System.out.println("1. Special Series");
        System.out.println("2. Fraction Product");

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

        switch(choice)
        {
            case 1:

                int n=0,s=0;

                for(int i=1;i<=5;i++)
                {
                    n=n*10+i;
                    s=s+n;
                }

                System.out.println("Sum = " + s);
                break;

            case 2:

                double p=1;

                for(int i=1;i<=9;i++)
                {
                    p=p*((double)i/(i+1));
                }

                System.out.println("Product = " + p);
                break;

            default:
                System.out.println("Invalid Choice");
        }
    }
}
```

Solve question number 11 12 13 14 15 16 17 18 ,sub questions also

11. Menu-Driven Series Program (It will be discussed in the final term)

11(a). Series Program

Find:

$s = x + x^2/2! + x^3/3! + ... n$

Answer

```
import java.util.*;

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

        int x,n,fact;
        double s=0;

        System.out.print("Enter value of x: ");
        x=sc.nextInt();

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

        for(int i=1;i<=n;i++)
        {
            fact=1;

            for(int j=1;j<=i;j++)
            {
                fact=fact*j;
            }

            s = s + (Math.pow(x,i)/fact);
        }

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

Explanation

Series:

$x + x^2/2! + x^3/3! + ...$

Factorial is calculated using inner loop.

11(b). Series Program

Find:

$s = 1/1^3 - 1/2^3 + 1/3^3 ..... 1/n^3$

Answer

```
import java.util.*;

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

        int n;
        double s=0;

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

```
        for(int i=1;i<=n;i++)
        {
            if(i%2==0)
                s = s - (1.0/Math.pow(i,3));
            else
                s = s + (1.0/Math.pow(i,3));
        }

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

---

12. Menu Driven Program using Switch Case (It will be discussed in the final term)

12(a). Series Program

Print:

```
0, 3, 7, 15, 24 .... n terms
```

Answer

```
import java.util.*;

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

        int n,term=0,diff=3;

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

        for(int i=1;i<=n;i++)
        {
            System.out.print(term + " ");

            term = term + diff;
            diff++;
        }
    }
}
```

Explanation

Difference increases:

```
+3,+4,+5,+6...
```

12(b). Sum of Series Program

Find:

```
S = 1/2 + 3/4 + 5/6 + 7/8 + .... + 19/20
```

Answer

```
class FractionSum
{
    public static void main(String args[])
    {
        double s=0;
```

```
        for(int i=1,j=2;i<=19;i=i+2,j=j+2)
        {
            s = s + ((double)i/j);
        }

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

---

13. Duck Number Program

Question

A Duck Number is a number that has at least one zero present in it.
Example:

```
3210, 7056, 8430709
```

Answer

```
import java.util.*;

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

        int n,d;
        boolean duck=false;

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

        while(n>0)
        {
            d=n%10;

            if(d==0)
            {
                duck=true;
                break;
            }

            n=n/10;
        }

        if(duck)
            System.out.println("Duck Number");
        else
            System.out.println("Not Duck Number");
    }
}
```

---

14. Menu Driven Program using Switch Statement (It will be discussed in the final term)

14(a). Factors of a Number

Question

Display all factors of a number excluding the number itself.
Example:

```
Input : 15
Output : 1,3,5
```

Answer

```
import java.util.*;

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

        int n;

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

        System.out.println("Factors are:");

        for(int i=1;i<n;i++)
        {
            if(n%i==0)
            {
                System.out.print(i + " ");
            }
        }
    }
}
```

14(b). Factorial Program

Question

Find factorial of a number.
Example:

```
5! = 1×2×3×4×5 = 120
```

Answer

```
import java.util.*;

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

        int n,fact=1;

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

        for(int i=1;i<=n;i++)
        {
            fact=fact*i;
        }

        System.out.println("Factorial = " + fact);
    }
}
```

---

15. Smallest Digit Program

Question

Write a program to input a number and find the smallest digit.
Sample Input:

```
6524
```

Output:

```
2
```

Answer

```
import java.util.*;

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

        int n,d,small=9;

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

        while(n>0)
        {
            d=n%10;

            if(d<small)
                small=d;

            n=n/10;
        }

        System.out.println("Smallest Digit = " + small);
    }
}
```

16. Menu Driven Program using Switch Statement(It will be discussed in the final term)

16(a). Fibonacci Series Program

Question

Generate first n terms of Fibonacci Series.
Series:
`0,1,1,2,3,5...`

Answer

```
import java.util.*;

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

        int n,a=0,b=1,c;

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

        System.out.print(a + " " + b + " ");

        for(int i=3;i<=n;i++)
        {
            c=a+b;

            System.out.print(c + " ");

            a=b;
            b=c;
```

```
        }
    }
}
```

Explanation

Each next term is:

```
previous two terms sum
```

16(b). Product of Even Digits Program

Question

Find the product of even digits.

Sample Input:

```
29485
```

Sample Output:

```
64
```

Answer

```
import java.util.*;

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

        int n,d,product=1;

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

        while(n>0)
        {
            d=n%10;

            if(d%2==0)
            {
                product=product*d;
            }

            n=n/10;
        }

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

---

17. Positive and Negative Number Program

Question

Write a program to:

1. Find number of positive numbers
2. Add all negative numbers

Answer

```
import java.util.*;

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

```
        int n,pos=0,negsum=0,num;

        System.out.print("How many numbers? ");
        n=sc.nextInt();

        for(int i=1;i<=n;i++)
        {
            System.out.print("Enter number: ");
            num=sc.nextInt();

            if(num>0)
                pos++;
            else if(num<0)
                negsum=negsum+num;
        }

        System.out.println("Positive Numbers = " + pos);
        System.out.println("Sum of Negative Numbers = " + negsum);
    }
}
```

---

18. Trimorphic Number Program

Question

A Trimorphic Number is a number whose cube ends with the same digit as the original number.
Example:
$6^3 = 216$
Last digit is 6.

Answer

```
import java.util.*;

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

        int n,cube;

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

        cube=n*n*n;

        if(cube%10==n)
            System.out.println("Trimorphic Number");
        else
            System.out.println("Not Trimorphic Number");
    }
}
```

Important Notes

Types of Loops

1. for loop
2. while loop
3. do-while loop

Jump Statements

- break
- continue

Infinite Loop Example

```
while(true)
{
}
```

****

Important Viva Questions

What is a Fibonacci Series?

A series in which each term is the sum of previous two terms.

What is a Duck Number?

A number containing at least one zero.

What is a Neon Number?

If sum of digits of square of number equals the number itself.

What is a Niven Number?

A number divisible by sum of its digits.

What is an Automorphic Number?

A number whose square ends with the same number.

What is a Trimorphic Number?

A number whose cube ends with the same digit as the original number.

Can you give me definitions some more type of numbers like a special number, auto morphic number, neon number etc . 20 more this type of numbers which has come in ICSE board or in the numeric handling questions search from the internet and give all those numbers with definition

Important Special Numbers in Java / ICSE Computer Applications

These number definitions are very important for:

- ICSE Board Exams
- Numeric Handling Programs
- Iterative Construct Programs
- Java Logic Building Questions

---

1. Prime Number

A number which is divisible only by 1 and itself.

Example

```
2, 3, 5, 7, 11
```

---

2. Composite Number

A number having more than two factors.

Example
```
4, 6, 8, 9
```

---

3. Even Number
A number divisible by 2.
Example
```
2, 4, 6, 8
```

---

4. Odd Number
A number not divisible by 2.
Example
```
1, 3, 5, 7
```

---

5. Armstrong Number
A number in which the sum of cubes of digits equals the number itself.
Example
```
153 = 1³ + 5³ + 3³
```

---

6. Neon Number
A number where the sum of digits of its square equals the number itself.
Example
```
9² = 81 → 8+1 = 9
```

---

7. Automorphic Number
A number whose square ends with the same digits as the number itself.
Example
```
25² = 625
```

---

8. Duck Number
A number containing at least one zero but not at the beginning.
Example
```
7056, 3210
```

---

9. Niven Number (Harshad Number)
A number divisible by the sum of its digits.
Example
```
18 → 1+8=9
18 divisible by 9
```

---

10. Disarium Number
A number where sum of digits raised to their positions equals the number.
Example
```
135 = 1¹ + 3² + 5³
```

---

11. Palindrome Number
A number that remains same after reversing digits.

Example

```
121, 1331
```

---

12. Special Number (Krishnamurthy Number)

A number where sum of factorial of digits equals the number.

Example

```
145 = 1! + 4! + 5!
```

---

13. Spy Number

A number where sum of digits equals product of digits.

Example

```
1124
1+1+2+4 = 8
1×1×2×4 = 8
```

---

14. Twisted Prime Number

A prime number whose reverse is also prime.

Example

```
13 → reverse 31
Both are prime
```

---

15. Twin Prime Number

Two prime numbers having difference of 2.

Example

```
11 and 13
17 and 19
```

---

16. Perfect Number

A number equal to the sum of its proper factors.

Example

```
6 = 1+2+3
```

---

17. Abundant Number

A number where sum of factors is greater than the number.

Example

```
12 → 1+2+3+4+6 = 16
```

---

18. Deficient Number

A number where sum of factors is less than the number.

Example

```
8 → 1+2+4 = 7
```

---

19. Happy Number

A number that finally becomes 1 after repeatedly adding squares of digits.

Example

```
19 → 82 → 68 → 100 → 1
```

---

20. Trimorphic Number

A number whose cube ends with the same digit as the number.

Example

```
6³ = 216
```

---

21. Pronic Number

A number which is product of two consecutive integers.

Example

```
12 = 3 × 4
20 = 4 × 5
```

---

22. Fibonacci Number

Numbers belonging to Fibonacci Series.

Example

```
0,1,1,2,3,5,8
```

---

23. Circular Prime Number

A prime number where all rotations are prime.

Example

```
197 → 971 → 719
```

---

24. Prime Palindrome Number

A number that is both prime and palindrome.

Example

```
131, 151
```

---

25. Unique Number

A number having no repeated digits.

Example

```
12345
```

---

26. Smith Number

A composite number where:
Sum of digits = Sum of digits of prime factors.

Example

```
666
```

---

27. Magic Number

A number whose repeated digit sum becomes 1.

Example

```
1729
1+7+2+9 = 19
1+9 = 10
1+0 = 1
```

---

28. Strong Number

A number where sum of factorial of digits equals the number itself.

Example

```
145
```

---

29. ISBN Number

A 10-digit number used for book identification.
Condition:

```
1×d1 + 2×d2 + ... + 10×d10
```

must be divisible by 11.

---

30. Amicable Numbers

Two numbers where sum of factors of one equals the other and vice versa.

Example

```
220 and 284
```

---

31. Strong Prime Number

A prime number greater than average of nearest prime numbers.

Example

```
17
```

---

32. Sunny Number

If next number is a perfect square.

Example

```
8+1 = 9
```

---

33. Adam Number

Square of number and square of reverse are reverses of each other.

Example

```
12² = 144
21² = 441
```

---

34. Emirp Number

A prime number whose reverse is another different prime.

Example

```
13 and 31
```

---

35. Kaprekar Number

If square split into two parts adds to original number.

Example

```
45² = 2025
20 + 25 = 45
```

Most Important Numbers for ICSE Board

Very Frequently Asked

1. Armstrong Number
2. Neon Number
3. Automorphic Number

4. Duck Number
5. Niven Number
6. Disarium Number
7. Special Number
8. Spy Number
9. Palindrome Number
10. Prime Number
11. Fibonacci Series
12. Trimorphic Number
13. ISBN Number
14. Twisted Prime Number
15. Happy Number

These are extremely important for:

- ICSE Class 9
- ICSE Class 10
- School practical exams
- Viva questions
- Logic building programs

****

Class 9 – (2025-26)
Computer Applications
Chapter 9: Iterative Constructs in Java

Solved Question Bank

A. Tick (✓) the correct answer.

1. Consider the following code snippet:

```
int i = 1, s = 0;
do {
   if (i % 4 == 0)
       i++;
   System.out.println(i * 2);
   i++;
} while (i <= 10);
```

Correct Answer: d. None of these
(Because it prints multiples of 2 but skips values when i % 4 == 0)

---

2. `for (int i = 10; i >= 1; i++)` is a/an:
   Correct Answer: c. Infinite
   (Since i++ increases value, condition i >= 1 will always remain true → infinite loop)

---

3. If "do-while" loop is exit-controlled, then "while" loop is:
   Correct Answer: b. Entry-controlled

---

4. What will the following program segment print?

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

Correct Answer: c. 6 : 4

---

5. What will be the output of the following code?

```
int m = 21, n = 15;
```

```
for (int i = 1; i < 5; i++) {
    m++;
    --n;
}
System.out.println("m = " + m);
System.out.println("n = " + n);
```

Correct Answer: b. m = 25, n = 11

B. Fill in the blanks (with answers).

1. **for** loops can have an empty loop body.
2. The two jump statements are **break** and **continue**.
3. To execute a loop 10 times: `for (i = 3; i <= 30; i = i + 3)`
4. `for (i = 10; i < 10; i++)` loop executes for **0** times.
5. An **entry-controlled** loop checks the condition at the time of entry.
6. Both **while** and **do-while** are suitable when the number of iterations is not known.

C. Short Answer Type Questions (Solved).

1. **Difference between multiline and documentation comment:**
   - Multiline: `/* comment */` → Used for general purpose comments.
   - Documentation: `/** comment */` → Used for JavaDoc documentation generation.
2. **Syntax to input short type using Scanner:**

```
Scanner sc = new Scanner(System.in);
short n = sc.nextShort();
```

5. **Three types of errors:**
   - Syntax errors
   - Runtime errors
   - Logical errors
6. **Logical error example:**

```
int a = 5, b = 0;
System.out.println(a/b); // Runtime error (divide by zero)
```

9. **Difference between try and catch:**
   - `try` contains risky code.
   - `catch` handles the exception if it occurs.

D. Programming Questions (Solved).

1. Series Programs

i. Print 1, 3, 5, 7 … 99

```
for(int i = 1; i <= 99; i += 2)
    System.out.print(i + " ");
```

ii. 20, 18, 16 … 2

```
for(int i = 20; i >= 2; i -= 2)
    System.out.print(i + " ");
```

iii. 2, 4, 8, 16 … 256

```
for(int i = 2; i <= 256; i *= 2)
    System.out.print(i + " ");
```

iv. 1/3, 2/6, 3/9 … 10/30

```
for(int i = 1; i <= 10; i++)
    System.out.print(i + "/" + (i*3) + " ");
```

v. 1, 12, 123, 1234, 12345

```
int num = 0;
for(int i = 1; i <= 5; i++) {
    num = num * 10 + i;
    System.out.print(num + " ");
}
```

vi. 1, 11, 111, 1111, 11111

```
int num = 0;
for(int i = 1; i <= 5; i++) {
    num = num * 10 + 1;
    System.out.print(num + " ");
}
```

2. Special Numbers

Neon Number

```
int n = 9;
int sq = n * n;
int sum = 0;
while(sq > 0) {
   sum += sq % 10;
   sq /= 10;
}
if(sum == n)
   System.out.println("Neon Number");
else
   System.out.println("Not Neon");
```

Palindrome

```
int n = 141, rev = 0, temp = n;
while(n > 0) {
   rev = rev * 10 + (n % 10);
   n /= 10;
}
if(temp == rev)
   System.out.println("Palindrome");
else
   System.out.println("Not Palindrome");
```

Disarium

```
int n = 135, temp = n, sum = 0, len = String.valueOf(n).length();
while(temp > 0) {
   int d = temp % 10;
   sum += Math.pow(d, len);
   len--;
   temp /= 10;
}
if(sum == n)
   System.out.println("Disarium Number");
else
   System.out.println("Not Disarium");
```

Automorphic

```
int n = 76, sq = n*n;
if(String.valueOf(sq).endsWith(String.valueOf(n)))
   System.out.println("Automorphic");
else
   System.out.println("Not Automorphic");
```

Duck Number

```
int n = 7056;
String s = String.valueOf(n);
if(s.contains("0"))
   System.out.println("Duck Number");
else
   System.out.println("Not Duck Number");
```

Krishnamurthy Number (145)

```
int n = 145, temp = n, sum = 0;
while(temp > 0) {
   int d = temp % 10, fact = 1;
   for(int i = 1; i <= d; i++) fact *= i;
   sum += fact;
   temp /= 10;
}
if(sum == n)
   System.out.println("Krishnamurthy Number");
else
   System.out.println("Not Krishnamurthy");
```

==========

Solved Programs (Without String Functions)

1. Print series (Math-based only)

i. 1, 3, 5, …, 99

```
for (int i = 1; i <= 99; i += 2) {
    System.out.print(i + " ");
}
```

ii. 20, 18, 16, …, 2

```
for (int i = 20; i >= 2; i -= 2) {
    System.out.print(i + " ");
}
```

iii. 2, 4, 8, …, 256

```
for (int i = 2; i <= 256; i *= 2) {
    System.out.print(i + " ");
}
```

iv. 1/3, 2/6, 3/9 … 10/30

```
for (int i = 1; i <= 10; i++) {
    System.out.print(i + "/" + (i*3) + " ");
}
```

v. 1, 12, 123, 1234, …

```
int num = 0;
for (int i = 1; i <= 5; i++) {
    num = num * 10 + i;
    System.out.print(num + " ");
}
```

vi. 1, 11, 111, 1111, …

```
int num = 0;
for (int i = 1; i <= 6; i++) {
    num = num * 10 + 1;
    System.out.print(num + " ");
}
```

---

2. Special Numbers (No String usage)

(a) Neon Number

(A number whose sum of digits of its square = number itself)

```
int n = 9;
int sq = n * n;
int sum = 0;
while (sq > 0) {
    sum += sq % 10;
    sq /= 10;
}
if (sum == n)
    System.out.println("Neon Number");
else
    System.out.println("Not Neon");
```

(b) Palindrome Number

(Reverse the digits and check equality)

```
int n = 141, rev = 0, temp = n;
while (temp > 0) {
    int d = temp % 10;
    rev = rev * 10 + d;
    temp /= 10;
}
if (rev == n)
    System.out.println("Palindrome");
else
    System.out.println("Not Palindrome");
```

(c) Disarium Number

(Sum of digits powered to their position = number)

```
int n = 135, temp = n, len = 0, sum = 0;

// Count number of digits
temp = n;
while (temp > 0) {
```

```
    len++;
    temp /= 10;
}

// Check Disarium
temp = n;
while (temp > 0) {
    int d = temp % 10;
    int pow = 1;
    for (int i = 1; i <= len; i++) {
        pow *= d;
    }
    sum += pow;
    len--;
    temp /= 10;
}

if (sum == n)
    System.out.println("Disarium Number");
else
    System.out.println("Not Disarium");
```

(d) Automorphic Number

(Square of number ends with the same digits as the number)

```
int n = 76, sq = n * n;
int temp = n;
int pow = 1;

// Find divisor (10, 100, 1000…)
while (temp > 0) {
    pow *= 10;
    temp /= 10;
}

// Compare last digits
if (sq % pow == n)
    System.out.println("Automorphic Number");
else
    System.out.println("Not Automorphic");
```

(e) Duck Number

(Has at least one 0, but not starting digit)

```
int n = 7056, temp = n;
boolean duck = false;

while (temp > 0) {
    int d = temp % 10;
    if (d == 0) {
        duck = true;
        break;
    }
    temp /= 10;
}

if (duck)
    System.out.println("Duck Number");
else
    System.out.println("Not Duck Number");
```

(f) Krishnamurthy Number (Strong Number)

(Sum of factorial of digits = number)

```
int n = 145, temp = n, sum = 0;

while (temp > 0) {
    int d = temp % 10;
    int fact = 1;
    for (int i = 1; i <= d; i++) {
        fact *= i;
```

```
    }
    sum += fact;
    temp /= 10;
}

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

(g) Niven Number (Harshad Number)
(Number divisible by sum of its digits)

```
int n = 111, temp = n, sum = 0;

while (temp > 0) {
    sum += temp % 10;
    temp /= 10;
}

if (n % sum == 0)
    System.out.println("Niven Number");
else
    System.out.println("Not Niven");
```

(h) Reverse Number and Absolute Difference

```
int n = 194, temp = n, rev = 0;

while (temp > 0) {
    rev = rev * 10 + (temp % 10);
    temp /= 10;
}

int diff = (n > rev) ? (n - rev) : (rev - n);
System.out.println("Reversed = " + rev);
System.out.println("Absolute Difference = " + diff);
```

Class 9 – Chapter 9: Iterative Constructs in Java

Section A: MCQs

1. Output of the program snippet

```
int i = 1, s = 0;
do {
    if (i % 4 == 0)
        i++;
    System.out.println(i * 2);
    i++;
} while (i <= 10);
```

☞ Output will not match any given exact sequence → **Answer: d. None of these**

---

2. for (i = 10; i >= 1; i++) loop is

Since `i++` with condition `i >= 1` never becomes false → **Answer: c. infinite**

---

3. If "do-while" loop is exit control loop, then "while" loop is

☞ **Answer: b. entry**

---

4. Program segment

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

☞ Output: `6:4` → **Answer: c. 6:4**

---

5. Code snippet (incomplete in your text but logically):
If syntax errors exist, it gives no output.

☞ **Answer: d. Not output**

Section B: Fill in the Blanks

1. **for** loops have an empty loop body.
2. The two jump statements are **break** and **continue**.
3. To execute a loop 10 times: `for (i = 1; i <= 10; i++).`
4. `for (i = 10; i < 10; i++)` loop executes for **0 times**.
5. An **entry** loop checks the condition at the time of entry.
6. Both **while** and **do-while** are suitable in situations where number of iterations is not known.

Section C: Short Answer Questions

1. Difference between multiline comment and documentation comment

- **Multiline comment**: `/* ... */` used to comment multiple lines, ignored by compiler.
- **Documentation comment**: `/** ... */` used to generate documentation via Javadoc.

2. Syntax to input a Short type value using Scanner

```
Scanner sc = new Scanner(System.in);
short n = sc.nextShort();
```

3. Three types of errors

- **Syntax errors** – Wrong language grammar.
- **Logical errors** – Wrong logic but compiles.
- **Runtime errors** – Errors while running (e.g., divide by zero).

4. Logical error with example

Example:

```
int a = 5, b = 10;
System.out.println("Average = " + (a + b) / 2.0); // Correct
// If written (a+b)/2 (integer division) → wrong logic
```

5. Difference between try and catch

- **try block**: contains statements that may cause exception.
- **catch block**: handles the exception if it occurs.

Section D: Java Programs (without String functions)

Q12. Series using switch case

```
import java.util.*;
class SeriesMenu {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.println("1. Series: 1,12,123,1234,12345");
        System.out.println("2. Series: 1/1 * 2/4 * 3/9 ... n terms");
        int ch = sc.nextInt();
        int n, i, num = 0;
        switch(ch) {
            case 1:
                for(i = 1; i <= 5; i++) {
                    num = num * 10 + i;
                    System.out.print(num + " ");
                }
                break;
            case 2:
                System.out.print("Enter n: ");
                n = sc.nextInt();
                double p = 1.0;
                for(i = 1; i <= n; i++) {
                    p *= (double)i / (i*i);
                }
                System.out.println("Product = " + p);
                break;
            default:
                System.out.println("Invalid choice");
        }
    }
```

```
}
```

Q13. Sum of series using switch case

```
import java.util.*;
class SeriesSum {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.println("1. S = x + x^2/2 + x^3/3 + ... + n terms");
        System.out.println("2. S = 1/1^3 - 1/2^3 + 1/3^3 ... 1/n^3");
        int ch = sc.nextInt();
        int n, i, x;
        switch(ch) {
            case 1:
                System.out.print("Enter x and n: ");
                x = sc.nextInt();
                n = sc.nextInt();
                double s1 = 0;
                for(i = 1; i <= n; i++) {
                    s1 += Math.pow(x, i) / i;
                }
                System.out.println("Sum = " + s1);
                break;
            case 2:
                System.out.print("Enter n: ");
                n = sc.nextInt();
                double s2 = 0;
                for(i = 1; i <= n; i++) {
                    if(i % 2 == 0)
                        s2 -= 1.0 / (i*i*i);
                    else
                        s2 += 1.0 / (i*i*i);
                }
                System.out.println("Sum = " + s2);
                break;
            default:
                System.out.println("Invalid choice");
        }
    }
}
```

Q14. Menu-driven program

```
import java.util.*;
class SeriesSwitch {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.println("1. Series: 0,3,7,15,24..n terms");
        System.out.println("2. Sum of series: 1/2 + 3/4 + 5/6 ... 19/20");
        int ch = sc.nextInt();
        int n, i;
        switch(ch) {
            case 1:
                System.out.print("Enter n: ");
                n = sc.nextInt();
                int term = 0;
                for(i = 1; i <= n; i++) {
                    term = (i*i - 1);
                    System.out.print(term + " ");
                }
                break;
            case 2:
                double sum = 0;
                for(i = 1; i <= 19; i+=2) {
                    sum += (double)i / (i+1);
                }
                System.out.println("Sum = " + sum);
                break;
            default:
                System.out.println("Invalid choice");
        }
```

```
    }
}
```

Q15. Duck Number

```
import java.util.*;
class DuckNumber {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt(), d, flag = 0, num = n;
        while(n > 0) {
            d = n % 10;
            if(d == 0) flag = 1;
            n /= 10;
        }
        if(flag == 1) System.out.println(num + " is Duck Number");
        else System.out.println(num + " is not Duck Number");
    }
}
```

Q16. Factors and Factorial (switch)

```
import java.util.*;
class FactorSwitch {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.println("1. Factors");
        System.out.println("2. Factorial");
        int ch = sc.nextInt();
        int n = sc.nextInt();
        switch(ch) {
            case 1:
                System.out.print("Factors: ");
                for(int i = 1; i < n; i++) {
                    if(n % i == 0) System.out.print(i + " ");
                }
                break;
            case 2:
                int f = 1;
                for(int i = 1; i <= n; i++) f *= i;
                System.out.println("Factorial = " + f);
                break;
            default:
                System.out.println("Invalid choice");
        }
    }
}
```

Q17. Find the smallest digit in a number

```
import java.util.*;
class SmallestDigit {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        int smallest = 9;
        int temp = n;
        while(temp > 0) {
            int digit = temp % 10;
            if(digit < smallest) smallest = digit;
            temp /= 10;
        }
        System.out.println("Smallest digit is " + smallest);
    }
}
```

Example:

Input: `6524` → Output: `2`

Q18. Check whether a number is prime palindrome

```
import java.util.*;
class PrimePalindrome {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        // Check palindrome
        int reversed = 0, temp = n;
        while(temp > 0) {
            reversed = reversed * 10 + temp % 10;
            temp /= 10;
        }
        boolean isPalindrome = (reversed == n);

        // Check prime
        boolean isPrime = true;
        if(n < 2) isPrime = false;
        for(int i = 2; i <= n/2; i++) {
            if(n % i == 0) {
                isPrime = false;
                break;
            }
        }

        if(isPalindrome && isPrime)
            System.out.println(n + " is a prime palindrome number");
        else
            System.out.println(n + " is not a prime palindrome number");
    }
}
```

Example:
Input: `131` → Output: `prime palindrome number`

Q19. Menu-driven Fibonacci series and product of even digits

```
import java.util.*;
class FibonacciEvenProduct {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.println("1. Fibonacci series");
        System.out.println("2. Product of even digits");
        int choice = sc.nextInt();

        switch(choice) {
            case 1:
                System.out.print("Enter n terms: ");
                int n = sc.nextInt();
                int a = 0, b = 1;
                System.out.print("Fibonacci series: " + a + " " + b + " ");
                for(int i = 3; i <= n; i++) {
                    int c = a + b;
                    System.out.print(c + " ");
                    a = b;
                    b = c;
                }
                System.out.println();
                break;
            case 2:
                System.out.print("Enter a number: ");
                int num = sc.nextInt();
                int product = 1, flag = 0;
                int temp = num;
                while(temp > 0) {
                    int digit = temp % 10;
                    if(digit % 2 == 0) {
                        product *= digit;
                        flag = 1;
                    }
```

```
                    temp /= 10;
                }
                if(flag == 0) product = 0; // no even digits
                System.out.println("Product of even digits = " + product);
                break;
            default:
                System.out.println("Invalid choice");
        }
    }
}
```

Q20. Count positive numbers and sum negative numbers

```
import java.util.*;
class PosNegNumbers {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of integers: ");
        int n = sc.nextInt();
        int positiveCount = 0;
        int negativeSum = 0;

        for(int i = 0; i < n; i++) {
            System.out.print("Enter number: ");
            int num = sc.nextInt();
            if(num > 0) positiveCount++;
            else if(num < 0) negativeSum += num;
        }
        System.out.println("Number of positive numbers = " + positiveCount);
        System.out.println("Sum of negative numbers = " + negativeSum);
    }
}
```

Q21. Check whether a number is Trimorphic

```
import java.util.*;
class TrimorphicNumber {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        int cube = n * n * n;
        int temp = n;
        int digits = 0;

        // Count number of digits
        while(temp > 0) {
            digits++;
            temp /= 10;
        }

        int divisor = 1;
        for(int i = 0; i < digits; i++) divisor *= 10;

        if(cube % divisor == n)
            System.out.println(n + " is a Trimorphic number");
        else
            System.out.println(n + " is not a Trimorphic number");
    }
}
```

Example:

Input: `6` → Output: `Trimorphic number` (since $6^3 = 216$, ends with 6)