Class 9 – 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)