

Control Statements in Java

Aim:

To understand and implement decision-making and looping control statements in Java.

PRE LAB EXERCISE

QUESTIONS

1. List different control statements in Java

Control statements are used to control the flow of execution of a program.

Types of Control Statements in Java

a) Selection (Decision-making) Statements

Used to make decisions based on conditions.

- if
- if–else
- else–if ladder
- switch

b) Iteration (Looping) Statements

Used to repeat a block of code.

- for
- while
- do–while

c) Jump (Branching) Statements

Used to transfer control from one part of the program to another.

- break

- continue
- return

2. Difference between for, while, and do-while loops

Feature	for loop	while loop	do-while loop
Condition check	Before loop starts	Before loop starts	After loop executes
Minimum execution	May not execute	May not execute	Executes at least once
Best used when	Number of iterations is known	Iterations not known	Loop must run once
Syntax complexity	Compact	Simple	Simple

Example:

for loop

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

while loop

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

do-while loop

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

3. What is the use of break and continue?

break statement

- Used to terminate a loop or switch statement immediately
- Control moves to the statement after the loop

Example:

```
for(int i = 1; i <= 5; i++) {  
    if(i == 3) {  
        break;  
    }  
    System.out.println(i);  
}
```

Output:

1
2

continue statement

- Used to skip the current iteration
- Control moves to the next iteration of the loop

Example:

```
for(int i = 1; i <= 5; i++) {  
    if(i == 3) {  
        continue;  
    }  
    System.out.println(i);  
}
```

Output:

1
2
4
5
✓

IN LAB EXERCISE

Objective:

To implement if-else and looping statements.

INPUT STATEMENT:

SCANNER CLASS

- ✓ The Scanner class in Java is used to read input from the user through the keyboard.
It is available in the package `java.util`.
- ✓ The Scanner object reads different types of input such as integer, float, double, and string and stores them in variables.
- ✓ To use the Scanner class, it must be imported before using it in the program.

SYNTAX:

- ✓ `Scanner sc = new Scanner(System.in);`

Commonly Used Scanner Methods:

- ✓ `nextInt()` – reads an integer value
- ✓ `nextFloat()` – reads a float value
- ✓ `nextDouble()` – reads a double value
- ✓ `next()` – reads a single word
- ✓ `nextLine()` – reads a complete line of text

PROGRAMS:

Program 1: Check Whether a Number is Positive

```
class PositiveNumber {  
    public static void main(String[] args) {  
        int n = 5;  
        if (n > 0) {  
            System.out.println("Positive Number");  
        }  
    }  
}
```

```
}
```

```
}
```

Output:

The screenshot shows a code editor interface with a left sidebar containing file icons for Python, Java, C/C++, SQL, Go, C, C++, JavaScript, and TypeScript. The main area displays a Java file named PositiveNumber.java. The code is as follows:

```
PositiveNumber.java
1 ~ class PositiveNumber {
2 ~     public static void main(String[] args) {
3 ~         int n = 5;
4 ~
5 ~         if (n > 0) {
6 ~             System.out.println("Positive Number");
7 ~         }
8 ~     }
9 ~
10 |}
11
12
13
14
```

Below the code, there are several buttons: Open, Save, Share, Run, and Clear. The "Run" button is highlighted in blue. To the right of the code, the output window shows the following text:

Positive Number
== Code Execution Successful ==

Program 2: Check Whether a Number is Even or Odd

```
class EvenOdd {
    public static void main(String[] args) {
        int n = 6;
        if (n % 2 == 0)
            System.out.println("Even Number");
        else
            System.out.println("Odd Number");
    }
}
```

}

Output:

The screenshot shows the Programiz Online Java Compiler interface. On the left, there's a sidebar with icons for various languages: Python, C/C++, C, C++, C#, Go, JS, and TS. The main area has tabs for "EvenOdd.java" and other file types like "Run", "Output", and "Clear". The code editor contains the following Java code:

```
1- class EvenOdd {  
2-     public static void main(String[] args) {  
3-         int n = 6;  
4-         if (n % 2 == 0)  
5-             System.out.println("Even Number");  
6-         else  
7-             System.out.println("Odd Number");  
8-     }  
9- }  
10-  
11-  
12-  
13-  
14-  
15-
```

The "Output" panel shows the execution results:

Even Number
==== Code Execution Successful ===

Program 3: Find Largest of Two Numbers

```
class LargestTwo {  
    public static void main(String[] args) {  
        int a = 10, b = 20;  
        if (a > b)  
            System.out.println("A is largest");  
        else  
            System.out.println("B is largest");  
    }  
}
```

Output:

The screenshot shows the Programiz Online Java Compiler interface. At the top, there's a promotional banner for a challenge: "Build your resume with HTML & CSS and win \$100" with the subtext "Get featured on Programiz PRO and the Wall of Inspiration." To the right of the banner is a "Join Challenge →" button. Below the banner, the Programiz logo and "Online Java Compiler" text are visible. On the far right, there's a "Programiz PRO" button. The main workspace consists of three panels: a code editor on the left containing Java code, a toolbar in the center with icons for copy, paste, run, share, and clear, and an output panel on the right displaying the execution results. The code editor contains the following Java code:

```
1- class LargestTwo {  
2- public static void main(String[] args) {  
3- int a = 10, b = 20;  
4- if (a > b)  
5- System.out.println("A is largest");  
6- else  
7- System.out.println("B is largest");  
8- }  
9- }  
10-  
11-  
12-  
13-  
14-
```

The output panel shows the execution results:

```
B is largest  
==== Code Execution Successful ===
```

Program 4: Grade Calculation

```
class Grade {  
public static void main(String[] args) {  
int marks = 75;  
if (marks >= 90)  
System.out.println("Grade A");
```

```
else if (marks >= 75)
System.out.println("Grade B");

else if (marks >= 50)
System.out.println("Grade C");

else
System.out.println("Fail");
}
```

}Output:

The screenshot shows the Programiz Online Java Compiler interface. On the left, there's a sidebar with various programming language icons: Python, C/C++, C, Go, Java (selected), JavaFX, Kotlin, Scala, and JS. The main area has tabs for 'Grade.java' and 'Output'. The code editor contains the following Java code:

```
1- class Grade {
2- public static void main(String[] args) {
3-     int marks = 75;
4-     if (marks >= 90)
5-         System.out.println("Grade A");
6-     else if (marks >= 75)
7-         System.out.println("Grade B");
8-     else if (marks >= 50)
9-         System.out.println("Grade C");
10-    else
11-        System.out.println("Fail");
12-    }
13- }
14- 
```

The 'Output' tab shows the result of running the code: "Grade B" followed by "==== Code Execution Successful ===".

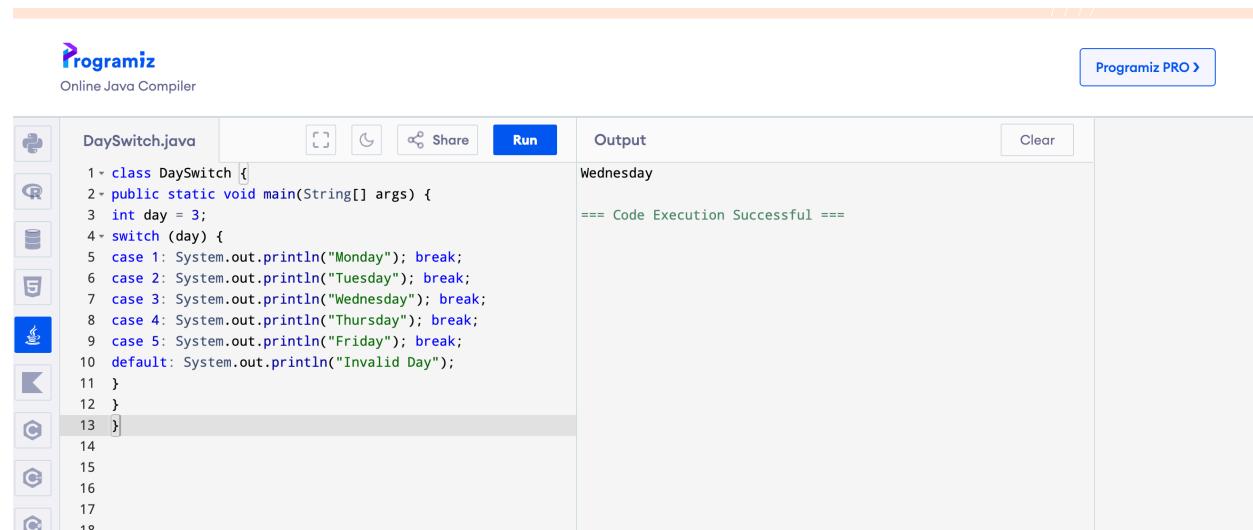
Program 5: Day of the Week

```
class DaySwitch {

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

```
int day = 3;  
switch (day) {  
    case 1: System.out.println("Monday"); break;  
    case 2: System.out.println("Tuesday"); break;  
    case 3: System.out.println("Wednesday"); break;  
    case 4: System.out.println("Thursday"); break;  
    case 5: System.out.println("Friday"); break;  
    default: System.out.println("Invalid Day");  
}  
}  
}
```

Output:



The screenshot shows the Programiz Online Java Compiler interface. On the left, there's a sidebar with various icons for file operations like Open, Save, Share, Run, and Help. The main area has tabs for 'DaySwitch.java' and 'Output'. The code editor contains the Java code provided above. The output window shows the result of running the program: 'Wednesday' followed by '== Code Execution Successful ==='. There are also 'Clear' and 'Programiz PRO' buttons at the top right.

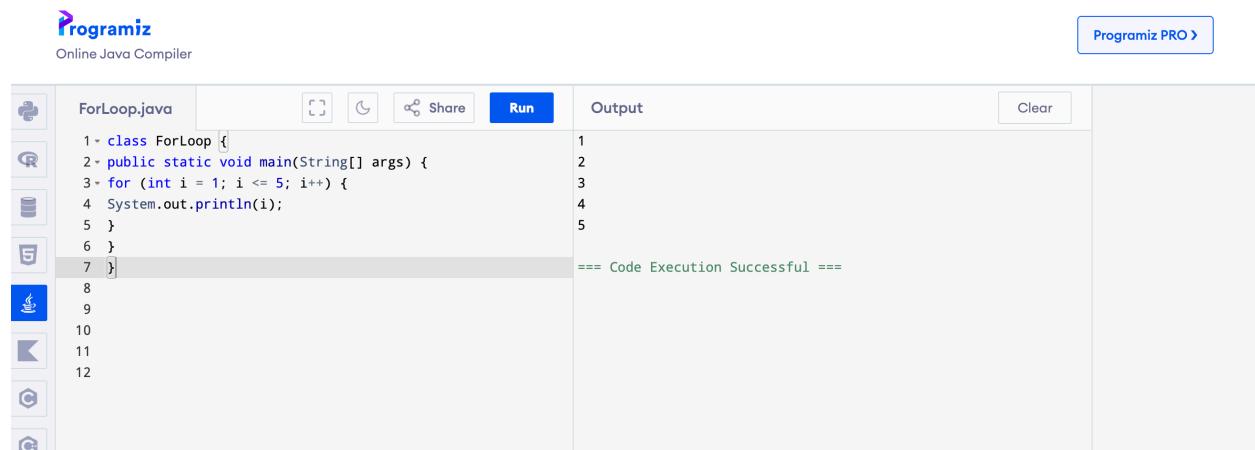
```
DaySwitch.java  
Run  
Output  
Clear  
Programiz PRO  
1 ~ class DaySwitch {  
2 ~     public static void main(String[] args) {  
3 ~         int day = 3;  
4 ~         switch (day) {  
5 ~             case 1: System.out.println("Monday"); break;  
6 ~             case 2: System.out.println("Tuesday"); break;  
7 ~             case 3: System.out.println("Wednesday"); break;  
8 ~             case 4: System.out.println("Thursday"); break;  
9 ~             case 5: System.out.println("Friday"); break;  
10 ~        default: System.out.println("Invalid Day");  
11 ~    }  
12 ~ }  
13 ~ }  
14 ~  
15 ~  
16 ~  
17 ~  
18 ~
```

Wednesday
== Code Execution Successful ===

Program 6: Print Numbers from 1 to 5

```
class ForLoop {  
    public static void main(String[] args) {  
        for (int i = 1; i <= 5; i++) {  
            System.out.println(i);  
        }  
    }  
}
```

Output:



The screenshot shows the Programiz Online Java Compiler interface. On the left, there's a sidebar with various icons for file operations like Open, Save, Share, Run, and Undo/Redo. The main area has tabs for 'ForLoop.java' and 'Output'. The code editor contains the Java code for printing numbers from 1 to 5. The output window shows the numbers 1, 2, 3, 4, and 5, followed by a message indicating successful execution.

Line	Content
1	class ForLoop {
2	public static void main(String[] args) {
3	for (int i = 1; i <= 5; i++) {
4	System.out.println(i);
5	}
6	}
7	}
8	
9	
10	
11	
12	

Output:
1
2
3
4
5
==== Code Execution Successful ===

Program 7: Print Numbers from 1 to 5

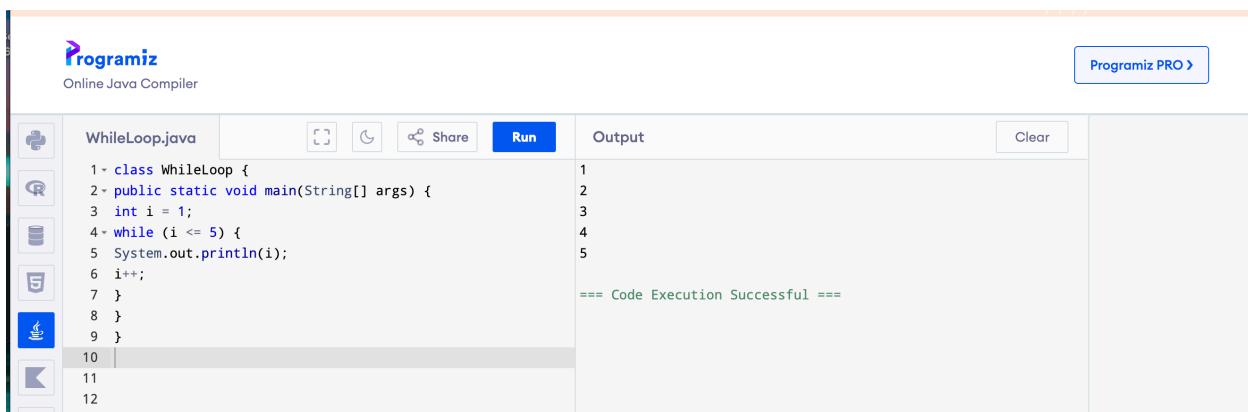
```
class WhileLoop {  
    public static void main(String[] args) {  
        int i = 1;  
        while (i <= 5) {  
            System.out.println(i);  
            i++;  
        }  
    }  
}
```

```
}
```

```
}
```

```
}
```

Output:



The screenshot shows the Programiz Online Java Compiler interface. On the left, there's a sidebar with various icons for file operations like Open, Save, and Print. The main area has tabs for "WhileLoop.java" and "Output". The code editor contains the following Java code:

```
1- class WhileLoop {  
2- public static void main(String[] args) {  
3- int i = 1;  
4- while (i <= 5) {  
5- System.out.println(i);  
6- i++;  
7- }  
8- }  
9- }  
10 |  
11  
12
```

The output window shows the numbers 1 through 5 printed sequentially, followed by a success message:

```
1  
2  
3  
4  
5  
==== Code Execution Successful ===
```

Program 8: Print Numbers from 1 to 5

```
class DoWhileLoop {  
public static void main(String[] args) {  
int i = 1;  
do {  
System.out.println(i);  
i++;  
}
```

```
} while (i <= 5);  
}  
}
```

Output:

Build your resume with HTML & CSS and win \$100
Get featured on Programiz PRO and the Wall of Inspiration.

Join Challenge →

Programiz PRO >

The screenshot shows a Java code editor with the file 'DoWhileLoop.java' containing a do-while loop that prints numbers from 1 to 5. The output window shows the printed numbers and a success message. The interface includes icons for file operations, a run button, and a share button.

Line	Content
1	class DoWhileLoop {
2	public static void main(String[] args) {
3	int i = 1;
4	do {
5	System.out.println(i);
6	i++;
7	} while (i <= 5);
8	}
9	
10	
11	
12	
13	

Output:
1
2
3
4
5
==== Code Execution Successful ===

Program 9: Sum of First 5 Natural Numbers

```
class SumNumbers {  
    public static void main(String[] args) {  
        int sum = 0;  
        for (int i = 1; i <= 5; i++) {  
            sum = sum + i;  
        }  
    }  
}
```

```

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

```

Output:

[Build your resume with HTML & CSS and win \\$100](#)
 Get featured on Programiz PRO and the Wall of Inspiration.
 [Join Challenge →](#)

Programiz
Online Java Compiler
Programiz PRO >

```

DoWhileLoop.java
1 ~ class DoWhileLoop {
2 ~     public static void main(String[] args) {
3 ~         int i = 1;
4 ~         do {
5 ~             System.out.println(i);
6 ~             i++;
7 ~         } while (i <= 5);
8 ~     }
9 ~
10
11
12
13

```

Run
Output
Clear

```

1
2
3
4
5
==== Code Execution Successful ====

```

Program 10: Multiplication Table of a Number

```

class MultiplicationTable {

public static void main(String[] args) {

int n = 5;

for (int i = 1; i <= 10; i++) {

System.out.println(n + " x " + i + " = " + (n * i));

```

```
}
```

```
}
```

```
}
```

Output:

Build your resume with HTML & CSS and win \$100
Get featured on Programiz PRO and the Wall of Inspiration.

Join Challenge →

Programiz PRO >

Programiz

Online Java Compiler

The screenshot shows a Java code editor interface. On the left, there's a sidebar with icons for file operations like new, open, save, and copy/paste. The main area has tabs for 'DoWhileLoop.java' and 'Output'. The code in 'DoWhileLoop.java' is:

```
1 class DoWhileLoop {  
2     public static void main(String[] args) {  
3         int i = 1;  
4         do {  
5             System.out.println(i);  
6             i++;  
7         } while (i <= 5);  
8     }  
9 }  
10  
11  
12  
13
```

The 'Run' button is highlighted in blue. To the right, the 'Output' tab shows the console output:

```
1  
2  
3  
4  
5  
==== Code Execution Successful ===
```

Postlab

1. What is the use of if statement?

Answer:

The if statement is used to check a condition and execute a block of code only if the condition is true.

2. Difference between if-else and else-if ladder

if-else	else-if ladder
Checks only two conditions	Checks multiple conditions
Has one if and one else	Has multiple else-if conditions
Executes either if or else	Executes only the first true condition
Used for simple decisions	Used for multiple choices

3. Why is switch statement used?

Answer:

The switch statement is used to select one block of code from multiple options based on the value of a variable.

Advantages:

- Cleaner than multiple if-else
- Improves readability
- Faster execution in some cases

4. Difference between for, while, and do-while loops

for loop	while loop	do-while loop
Entry-controlled	Entry-controlled	Exit-controlled
Condition checked first	Condition checked first	Condition checked after execution
Used when iterations are known	Used when iterations are unknown	Used when loop must run at least once

5. Which loop executes at least once?

Answer:

The do-while loop executes at least once, even if the condition is false.

Reason:

- The condition is checked after executing the loop body.

Result:

Thus the different control statements were executed successfully with expected output.

ASSESSMENT

Description	Max Marks	Marks Awarded
Pre Lab Exercise	5	
In Lab Exercise	10	
Post Lab Exercise	5	
Viva	10	
Total	30	
Faculty Signature		