

CONTROL STRUCTURES



Lesson Objectives

By the end of this lesson, students should be able to:

- Define and explain control structures in programming and algorithms.
 - Write pseudocode using **selection** and **iteration** structures.
 - Apply control structures to solve simple problems.
 - Convert algorithms into flowcharts.
-

1. Introduction to Control Structures

In any program or algorithm, instructions are not always executed in a straight line. Sometimes, we need to make **decisions** or **repeat actions** — this is where *control structures* come in.

Control structures determine **how the flow of control moves** in an algorithm.

Types of Control Structures

1. **Sequential:** Steps executed one after another.
 2. **Selection (Decision-making):** Choose between different actions.
 3. **Iteration (Repetition):** Repeat certain actions multiple times.
-

2. Selection Structures

Selection allows the algorithm to make **choices based on conditions**.

A. IF Statement

Used when an action should occur only if a condition is true.

Syntax:

```
IF condition THEN  
    statement(s)  
ENDIF
```

Example:

```
IF age >= 18 THEN  
    PRINT "You are eligible to vote"  
ENDIF
```



B. IF...ELSE Statement

Used when there are **two possible outcomes**.

Syntax:

```
IF condition THEN
    statement(s)
ELSE
    statement(s)
ENDIF
```

Example:

```
IF score >= 50 THEN
    PRINT "Pass"
ELSE
    PRINT "Fail"
ENDIF
```

C. Nested IF

Used when multiple conditions must be checked **in sequence**.

Example:

```
IF score >= 70 THEN
    PRINT "Excellent"
ELSE
    IF score >= 50 THEN
        PRINT "Good"
    ELSE
```

```
    PRINT "Fail"
ENDIF
ENDIF
```

D. CASE (or SWITCH) Statement

Used when there are **many alternatives**.

Syntax:

```
CASE variable OF
    value1: statement(s)
    value2: statement(s)
    ...
    OTHERWISE: statement(s)
ENDCASE
```



Example:

```
CASE grade OF
    "A": PRINT "Excellent"
    "B": PRINT "Good"
    "C": PRINT "Average"
    OTHERWISE: PRINT "Fail"
ENDCASE
```

3. Iteration Structures

Iteration is used when a task must be **repeated** until a condition is met.

A. FOR Loop

Used when the number of repetitions is known.

Syntax:

```
FOR counter = start TO end  
    statement(s)  
NEXT counter
```

Example:

```
FOR i = 1 TO 5  
    PRINT i  
NEXT i
```



B. WHILE Loop

Executes while a condition is **true**.

Syntax:

```
WHILE condition DO
```

```
    statement(s)
ENDWHILE
```

Example:

```
x = 1
WHILE x <= 5 DO
    PRINT x
    x = x + 1
ENDWHILE
```

C. REPEAT...UNTIL Loop

Executes the block **at least once** and continues until the condition becomes true.

Syntax:

```
REPEAT
    statement(s)
UNTIL condition
```

Example:

```
REPEAT
    INPUT number
UNTIL number > 0
```

4. Example: Find the Largest Number Among Three Values

Problem:

Write pseudocode to find the largest of three numbers A, B, and C.

Pseudocode:

INPUT A, B, C

IF $A \geq B$ AND $A \geq C$ THEN

 largest = A

ELSE IF $B \geq A$ AND $B \geq C$ THEN

 largest = B

ELSE

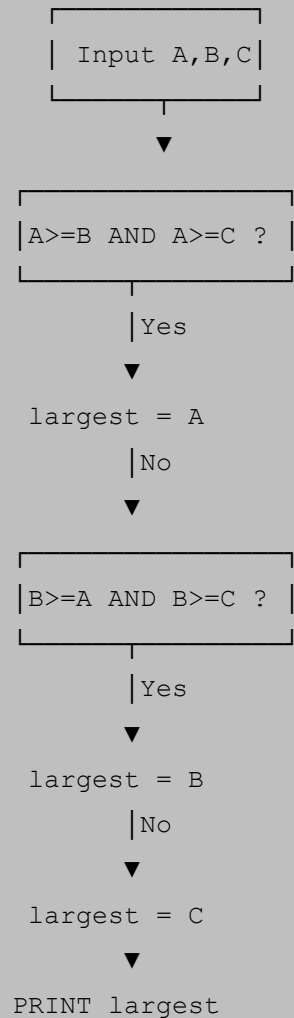
 largest = C

ENDIF

PRINT "The largest number is ", largest

Flowchart Representation

Below is a simplified version of the logic flow (describe or draw in class):



5. Activity: Design a Multiplication Table Algorithm

Task:

Write pseudocode that displays the multiplication table for a number entered by the user (from 1 to 12).

Pseudocode:

```
INPUT number
FOR i = 1 TO 12
```



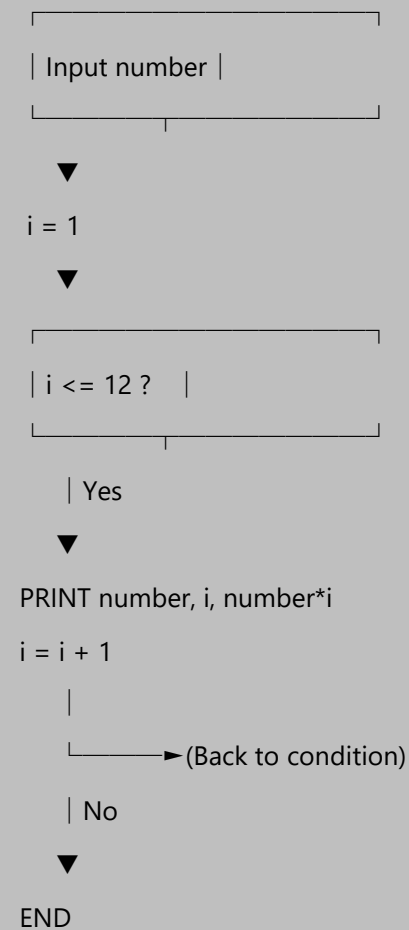
```
    PRINT number, " x ", i, " = ", number * i
NEXT i
```

Sample Output (for 5):

```
5 x 1 = 5
5 x 2 = 10
...
5 x 12 = 60
```

Flowchart for Multiplication Table

You can illustrate it as:



6. Exercises

1. Write pseudocode to determine if a number is **positive, negative, or zero**.
 2. Write pseudocode to find the **sum of all even numbers** between 1 and 20.
 3. Write pseudocode that asks for a user's score and prints a **grade** using CASE.
 4. Write pseudocode that calculates the **factorial** of a given number.
 5. Write pseudocode to print all **odd numbers from 1 to 15** using a WHILE loop.
-

