G.C.E. O/L Grade 11 ICT

Lesson 1: Programming

- 1.1 Problem Analysis: Inputs, Process and Outputs
- 1.2 Algorithms: Sequence, Selection and Iteration
- 1.3 Flowcharts and Pseudocode
- 1.4 Identifiers, Variables, Constants and Data Types
- 1.5 Operators: Arithmetic, Comparison and Logical
- 1.6 Selection: If–EndIf, If–Else–EndIf and Case
- 1.7 Iteration: FOR, WHILE and REPEAT–UNTIL Loops
- 1.8 Nested Control Structures
- 1.9 One-Dimensional Arrays
- 1.10 Sub-programs: Functions and Procedures
- 1.11 Programming Languages, Translators and Debugging

© 2026 schoolict.net (Ruwan Mangala Suraweera)
All rights reserved.

G.C.E. O/L Grade 11 ICT · Lesson 1

1.1 Problem Analysis: Inputs, Process and Outputs

Syllabus 10.1. Before we write a program, we must understand the problem. A **program** is a set of instructions that tells a computer what to do, step by step. In this note you will learn to find the **inputs**, the **outputs** and the **process** of a problem, to think of **different ways** to solve it, and to choose the **best** one. This is called **problem analysis**.

1. What is programming?

A computer cannot think by itself. It only follows instructions. Writing those instructions in a language the computer understands is called **programming**, and the person who writes them is a **programmer**. A program can be small (adding two numbers) or very big (the Sri Lanka Railways e-ticket system, a bank ATM or a mobile game).

> 💡 Think of a recipe for kiribath. The rice, coconut milk and salt are the **inputs**; cooking is the **process**; the plate of kiribath is the **output**. A program works the same way.

2. Steps of problem solving

Understand the problem
ගැටලුව තේරුම් ගන්න

Find inputs and outputs
ආදාන සහ ප්‍රතිදාන සොයන්න

Plan the process
ක්‍රියාවලිය සැලසුම් කරන්න

Choose the best solution
හොඳම විසඳුම තෝරන්න

Write and test the program
වැඩසටහන ලියා පරීක්ෂා කරන්න

3. Inputs, process and outputs (IPO)

Input – data given to the program (typed, scanned, read from a sensor). **Process** – the work done on the data (calculate, compare, sort). **Output** – the result given back (shown on screen, printed, sent as an SMS). We often write them in an **IPO table**.

Problem	Input	Process	Output
Shop bill	Price, quantity	Total = Price × Quantity	Total
Average of 3 marks	Mark1, Mark2, Mark3	Add, then divide by 3	Average
Electricity bill (CEB)	Units used	Units × rate + fixed charge	Amount to pay
Pass or fail	Mark	Is mark ≥ 35?	“Pass” or “Fail”

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

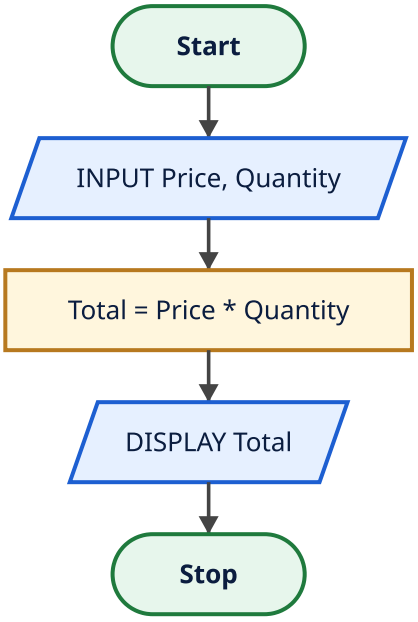
Bus ticket (Colombo–Kandy)	Number of passengers	Passengers × fare	Total fare

4. A worked example: the shop bill

Problem: A shop sells soap at Rs. 150 each. Find the total for any number of soaps. **Input:** price, quantity. **Process:** total = price × quantity. **Output:** total. Below is the same solution as pseudocode, as a flowchart and as a real program. Don't worry about the code yet – you will learn it step by step in 1.3 to 1.10.

Pseudocode

```
BEGIN
  INPUT Price, Quantity
  Total = Price * Quantity
  DISPLAY Total
END
```



Flowchart – Shop bill: price × quantity

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Shop bill: price × quantity

Pascal

```
program ShopBill;
var
  price, qty, total: integer;
begin
  write('Enter price (Rs.): ');
  readln(price);
  write('Enter quantity: ');
  readln(qty);
  total := price * qty;
  writeln('Total = Rs. ', total);
end.
```

Python

```
price = int(input("Enter price (Rs.): "))
qty = int(input("Enter quantity: "))
total = price * qty
print("Total = Rs.", total)
```

Sample run

```
Enter price (Rs.): 150
Enter quantity: 5
Total = Rs. 750
```

5. Alternative solutions (the solution space)

Most problems can be solved in **more than one way**. All the possible ways together are called the **solution space**. Example: find the sum of the numbers 1 to 100. **Method 1:** add them one by one with a loop (100 additions). **Method 2:** use the formula N × (N + 1) ÷ 2 (one multiplication and one division). Both give **5050**, but Method 2 is much faster.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Method 1: add the numbers one by one

Pascal

```
program SumLoop;
var
  i, total: integer;
begin
  total := 0;
  for i := 1 to 100 do
    total := total + i;
  writeln('Sum = ', total);
end.
```

Python

```
total = 0
for i in range(1, 101):
    total = total + i
print("Sum =", total)
```

Sample run

```
Sum = 5050
```

Method 2: use the formula N × (N + 1) ÷ 2

Pascal

```
program SumFormula;
var
  n, total: integer;
begin
  n := 100;
  total := n * (n + 1) div 2;
  writeln('Sum = ', total);
end.
```

Python

```
n = 100
total = n * (n + 1) // 2
print("Sum =", total)
```

Sample run

```
Sum = 5050
```

Compare	Method 1 (loop)	Method 2 (formula)
Number of steps for N = 100	About 100 additions	2 operations
Easy to understand?	Yes	Yes, if you know the formula
Speed for N = 1 000 000	Slower	Very fast
Result	5050	5050

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

6. How to choose the best solution

- **Correct** – gives the right answer for every valid input.
- **Efficient** – fewer steps, less time and memory.
- **Simple** – easy to read, test and change later.
- **Cost and resources** – uses the hardware and software we have.

NEW 2026 2026: You can practise programming free on a school computer or even a phone. Python and Pascal both run in **online compilers** (for example OnlineGDB or Programiz) – no installation needed. The programs on these pages were all really run, and the sample outputs are copied from the real results.

Key Words

Program – A set of instructions a computer follows step by step.	**Programming** – Writing instructions for a computer in a programming language.
Programmer – A person who writes programs.	**Problem analysis** – Studying a problem to find its inputs, process and outputs.
Input – Data given to a program.	**Process** – The work done to change inputs into outputs.
Output – The result a program gives.	**IPO table** – A table listing the inputs, process and outputs of a problem.
Solution space – All the possible ways to solve a problem.	**Alternative solution** – A different way of solving the same problem.
Efficient – Uses fewer steps, less time and less memory.	**Online compiler** – A website where you can type and run a program without installing anything.

Sample Q&A

1. What is a program?

Answer: A set of instructions that tells a computer what to do, step by step.

2. Name the three parts we find in problem analysis.

Answer: Input, process and output.

3. Give the inputs and output of a program that finds the area of a rectangle.

Answer: Inputs: length and width. Output: area (process: area = length × width).

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

4. What is the solution space?

> **Answer:** The set of all possible ways to solve a problem.

5. Give two ways to find the sum of 1 to 100 and say which is better.

> **Answer:** Add one by one with a loop, or use N × (N + 1) ÷ 2. The formula is better – far fewer steps.

6. List three things to check when choosing the best solution.

> **Answer:** It is correct, efficient (fast, little memory) and simple to read and change.

MCQ

1. A set of instructions that a computer follows is called a...

A. program
B. network
C. database table
D. monitor

2. In a shop bill program, the quantity bought is...

A. an output
B. a printer
C. an input
D. a process

3. In the same program, the total amount shown on the screen is...

A. an input
B. a variable name only
C. an output
D. a reserved word

4. “Total = Price × Quantity” is part of the...

A. process
B. output
C. hardware
D. input

5. The first step of problem solving is to...

A. buy a computer
B. understand the problem
C. type the program
D. print the output

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

6. For a program that finds the average of three marks, the inputs are…

A. the average
B. the printer
C. the number 3 only
D. the three marks

7. All the possible ways of solving a problem are together called the…

A. output space
B. input device
C. solution space
D. memory space

8. Using N × (N + 1) ÷ 2, the sum of 1 to 100 is…

A. 10100
B. 5050
C. 100
D. 5000

9. Why is the formula method better than adding 1 to 100 one by one?

A. It needs far fewer steps
B. It gives a different answer
C. It uses more memory
D. It needs a printer

10. A good solution should be…

A. long and complex
B. impossible to change
C. correct, efficient and simple
D. correct only for one input

11. In a CEB electricity bill program, the number of units used is…

A. an output
B. the solution space
C. a constant output
D. an input

12. Which is an output device that can show a program’s result?

A. Monitor
B. Keyboard
C. Mouse
D. Scanner

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

13. The table that lists inputs, process and outputs is the…

A. truth table
B. IPO table
C. trace table only
D. periodic table

14. For price 150 and quantity 5, the shop bill program shows…

A. Total = Rs. 155
B. Total = Rs. 145
C. Total = Rs. 30
D. Total = Rs. 750

15. You can run a Python or Pascal program without installing anything by using…

A. a spreadsheet chart
B. an online compiler
C. a word processor
D. a scanner

Structured Questions

1. (i) What is problem analysis? (1 mark)
(ii) Draw an IPO table for a program that finds the area of a rectangle. (3 marks) *(4 marks)*

> (i) Studying a problem to find its inputs, process and outputs before writing the program. (ii) Input: length, width (1); Process: area = length × width (1); Output: area (1).

2. A bus from Colombo to Kandy charges the same fare for every passenger. Identify the inputs, process and output of a program that finds the total fare. (3 marks) *(3 marks)*

> Inputs: number of passengers and fare (1); process: total = passengers × fare (1); output: total fare (1).

3. Give two different methods to find the sum of the numbers 1 to N. Which is more efficient? Why? (4 marks) *(4 marks)*

> Method 1: add 1, 2, 3 … N one by one with a loop (1). Method 2: use N × (N + 1) ÷ 2 (1). Method 2 is more efficient (1) because it uses only two operations whatever N is, while the loop needs N additions (1).

4. List the five steps of solving a problem with a computer, in order. (5 marks) *(5 marks)*

> 1 Understand the problem; 2 find inputs and outputs; 3 plan the process (algorithm); 4 choose the best solution; 5 write and test the program (1 each).

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

5. Give three things you should check when choosing the best solution from the solution space. (3 marks) *(3 marks)*

> Correct for all valid inputs; efficient (fewer steps, less time/memory); simple to read, test and change; suits the available resources (any three).

Summary

- Program = set of instructions for a computer.
- Problem analysis comes before writing code.
- Input = data in; output = result out.
- Process = work that changes inputs into outputs.
- Use an IPO table to plan.
- Solution space = all possible solutions.
- Best solution: correct, efficient, simple.
- Sum 1–100: loop or formula; both give 5050.

MCQ answer key – 1.1

1. **A** – A program is a set of instructions.
2. **C** – It is data given to the program.
3. **C** – It is the result.
4. **A** – Calculation = process.
5. **B** – Understand first.
6. **D** – Marks are given; average is the output.
7. **C** – Solution space.
8. **B** – 100 × 101 ÷ 2 = 5050.
9. **A** – Same answer, fewer steps.
10. **C** – Correct, efficient, simple.
11. **D** – Units are read from the meter and given to the program.
12. **A** – Monitor shows output.
13. **B** – IPO = Input-Process-Output.
14. **D** – 150 × 5 = 750.
15. **B** – Online compilers run code in the browser.

© 2026 schoolict.net (Ruwan Mangala Suraweera). All rights reserved.

G.C.E. O/L Grade 11 ICT · Lesson 1

1.2 Algorithms: Sequence, Selection and Iteration

Syllabus 10.2. An **algorithm** is a list of clear steps, in the correct order, that solves a problem. Every algorithm – from making tea to running a bank – is built from just **three control structures**: **sequence**, **selection** and **iteration**. Learn these three and you can plan any program.

1. What is an algorithm?

An algorithm is a **step-by-step plan**. It is not yet a program – it is written in simple words, pseudocode or a flowchart. A good algorithm has: a clear **start and end**, steps that are **clear** (no guessing), steps in the **correct order**, and it **finishes** after a limited number of steps.

> 💡 Everyday algorithm – making a cup of tea: 1 Boil water. 2 Put a tea bag in the cup. 3 Pour the hot water. 4 If you like sugar, add sugar. 5 Stir. Step 4 is a **selection**. If you stir “until the sugar dissolves”, that is an **iteration**.

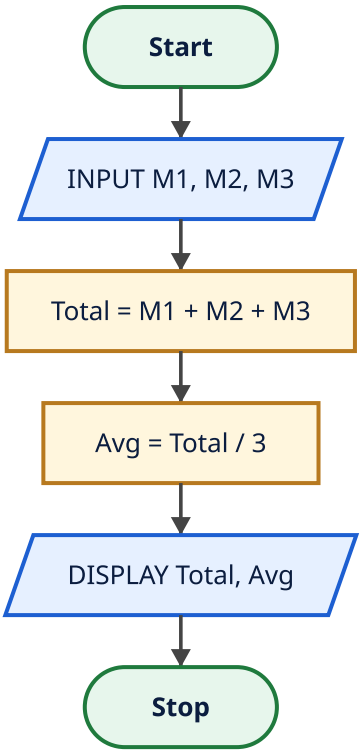
2. Sequence

In a **sequence**, the steps run **one after another**, from top to bottom, each step exactly once. Example: input three marks, find the total, find the average, display them.

Pseudocode

```
BEGIN
  INPUT M1, M2, M3
  Total = M1 + M2 + M3
  Avg = Total / 3
  DISPLAY Total, Avg
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – Sequence: total and average of three marks

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Sequence: total and average of three marks

Pascal

```
program Average3;
var
  m1, m2, m3, total: integer;
  avg: real;
begin
  write('Enter mark 1: ');
  readln(m1);
  write('Enter mark 2: ');
  readln(m2);
  write('Enter mark 3: ');
  readln(m3);
  total := m1 + m2 + m3;
  avg := total / 3;
  writeln('Total = ', total);
  writeln('Average = ', avg:0:2);
end.
```

Python

```
m1 = int(input("Enter mark 1: "))
m2 = int(input("Enter mark 2: "))
m3 = int(input("Enter mark 3: "))
total = m1 + m2 + m3
avg = total / 3
print("Total =", total)
print(f"Average = {avg:.2f}")
```

Sample run

```
Enter mark 1: 75
Enter mark 2: 68
Enter mark 3: 72
Total = 215
Average = 71.67
```

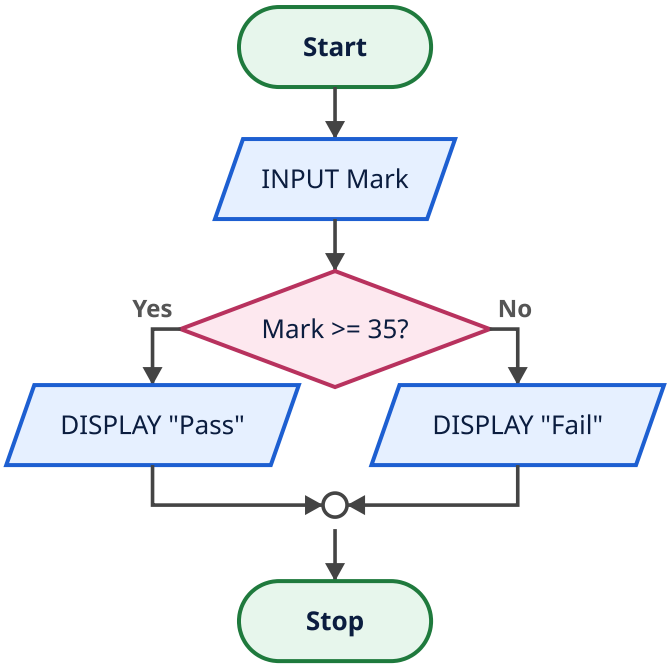
3. Selection

In **selection**, the computer checks a **condition** (a question with a Yes/No answer) and **chooses** which steps to run. Example: if the mark is 35 or more, display “Pass”, otherwise display “Fail”. Selection is also called a **decision**.

Pseudocode

```
BEGIN
  INPUT Mark
  IF Mark >= 35 THEN
    DISPLAY "Pass"
  ELSE
    DISPLAY "Fail"
  ENDIF
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – Selection: Pass or Fail

Selection: Pass or Fail

Pascal

```
program PassFail;
var
  mark: integer;
begin
  write('Enter mark: ');
  readln(mark);
  if mark >= 35 then
    writeln('Pass')
  else
    writeln('Fail');
end.
```

Python

```
mark = int(input("Enter mark: "))
if mark >= 35:
    print("Pass")
else:
    print("Fail")
```

Sample run 1

```
Enter mark: 52
Pass
```

Sample run 2

```
Enter mark: 28
Fail
```

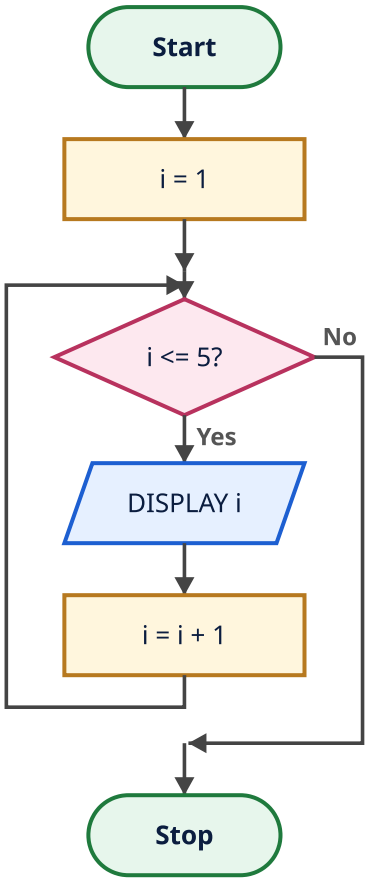
© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

4. Iteration

In **iteration** (also called **repetition** or a **loop**), some steps are **repeated** again and again while a condition is true, or a fixed number of times. Example: display the numbers 1 to 5. Without a loop we would need five DISPLAY steps; with a loop we need only one.

Pseudocode

```
BEGIN
  FOR i = 1 TO 5
    DISPLAY i
  ENDFOR
END
```



Flowchart – Iteration: display 1 to 5

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Iteration: display 1 to 5

Pascal

```
program Count5;
var
  i: integer;
begin
  for i := 1 to 5 do
    writeln(i);
end.
```

Python

```
for i in range(1, 6):
    print(i)
```

Sample run

```
1
2
3
4
5
```

5. The three structures side by side

Structure	What happens	Pseudocode words	Daily-life example
Sequence	Steps run in order, once each	INPUT, assign, DISPLAY	Wake up, wash, eat breakfast
Selection	A condition chooses a path	IF … THEN … ELSE … ENDIF	If it rains, take an umbrella
Iteration	Steps repeat	FOR, WHILE, REPEAT … UNTIL	Run 4 rounds of the school ground

NEW 2026 2026 – **Scratch and block coding.** In Scratch (scratch.mit.edu) you build programs by snapping blocks together. The same three structures are there: blocks stacked one under another are a **sequence**, the "if … then … else" block is **selection**, and "repeat 10" or "repeat until" is **iteration**. Scratch is a great way to practise algorithms before typing Pascal or Python.

Key Words

Algorithm – A clear, ordered list of steps that solves a problem.

Control structure – A way of controlling the order in which steps run.

Sequence – Steps run one after another, each once.

Selection – Choosing which steps to run using a condition.

Iteration – Repeating a group of steps.

Loop – Another name for iteration.

Condition – A question whose answer is True (Yes) or False (No).

Decision – Another name for selection.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Finite – An algorithm must end after a limited number of steps.

Average – Total divided by the number of values.

Scratch – A free block-based programming tool.

Block coding – Programming by joining blocks instead of typing code.

Sample Q&A

1. What is an algorithm?

> **Answer:** A clear list of steps, in the correct order, that solves a problem and ends.

2. Name the three control structures.

> **Answer:** Sequence, selection and iteration.

3. Which structure is used to decide Pass or Fail?

> **Answer:** Selection.

4. Which structure is used to display 1 to 100?

> **Answer:** Iteration (a loop).

5. For marks 75, 68 and 72, what are the total and average?

> **Answer:** Total = 215, average = 71.67.

6. Give two qualities of a good algorithm.

> **Answer:** Clear steps; correct order; a clear start and end; it finishes (any two).

MCQ

1. A clear, ordered list of steps that solves a problem is a/an…

A. operating system
B. algorithm
C. compiler
D. database

2. Steps that run one after another, each once, form a…

A. iteration
B. selection
C. loop
D. sequence

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

3. “If it rains, take an umbrella” is an example of…

A. selection
B. iteration
C. sequence
D. an array

4. “Run 4 rounds of the ground” is an example of…

A. sequence only
B. iteration
C. a constant
D. selection

5. Which is NOT one of the three control structures?

A. Compilation
B. Sequence
C. Selection
D. Iteration

6. A condition always has an answer of…

A. a name
B. a whole number
C. True or False
D. a picture

7. The total of the marks 75, 68 and 72 is…

A. 215
B. 205
C. 71.67
D. 225

8. For a mark of 28, the Pass/Fail program displays…

A. 28
B. Pass
C. Nothing
D. Fail

9. Another name for iteration is…

A. constant
B. loop
C. decision
D. output

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

10. Another name for selection is…

A. decision
B. loop
C. variable
D. sequence

11. An algorithm must end after a limited number of steps. This means it is…

A. infinite
B. hidden
C. finite
D. random

12. The count5 program displays how many numbers?

A. 4
B. 1
C. 5
D. 6

13. In Scratch, the “repeat 10” block is…

A. an output device
B. iteration
C. a variable
D. selection

14. In Scratch, the “if … then … else” block is…

A. an array
B. sequence
C. iteration
D. selection

15. Which of these is written BEFORE the program, to plan it?

A. The output
B. The compiler
C. An algorithm
D. The error message

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Structured Questions

1. (i) Define an algorithm. (1 mark)
(ii) Name the three control structures and give one daily-life example of each. (6 marks) *(7 marks)*

> (i) A clear, ordered, finite list of steps that solves a problem. (ii) Sequence – get ready for school step by step; selection – if it rains take an umbrella; iteration – run 4 rounds (1 name + 1 example each).

2. Write an algorithm in simple steps to find the area of a rectangle. Which control structure does it use? (4 marks) *(4 marks)*

> 1 Start. 2 Input length and width. 3 Area = length × width. 4 Display area. 5 Stop (3). It uses sequence (1).

3. Write steps to display "Pass" if a mark is 35 or more and "Fail" otherwise. Which structure is this? (3 marks) *(3 marks)*

> Input mark; if mark ≥ 35 then display "Pass" else display "Fail" (2). Selection (1).

4. Explain why a loop is better than writing DISPLAY 100 times to show the numbers 1 to 100. (2 marks) *(2 marks)*

> The loop needs only a few lines (1), is easier to change (e.g. to 1–1000) and has fewer mistakes (1).

5. Write the steps for making a cup of tea. Mark one step that is selection and one that could be iteration. (4 marks) *(4 marks)*

> Boil water; put tea bag in cup; pour water; if sugar is wanted add sugar (selection); stir until sugar dissolves (iteration) (2 for steps, 1 + 1 for marking).

Summary

- Algorithm = clear, ordered, finite steps.
- Three control structures: sequence, selection, iteration.
- Sequence: steps one after another.
- Selection: a condition chooses the path.
- Iteration: steps repeat (loop).
- A condition is True or False.
- Plan the algorithm before coding.
- Scratch blocks use the same three structures.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

MCQ answer key – 1.2

1. **B** – Definition of algorithm.
2. **D** – Sequence.
3. **A** – A condition chooses.
4. **B** – Repeated 4 times.
5. **A** – Compilation is translating code.
6. **C** – Yes/No = True/False.
7. **A** – 75+68+72 = 215.
8. **D** – 28 < 35.
9. **B** – Loop / repetition.
10. **A** – Decision.
11. **C** – Finite.
12. **C** – 1, 2, 3, 4, 5.
13. **B** – It repeats.
14. **D** – It chooses.
15. **C** – Plan first.

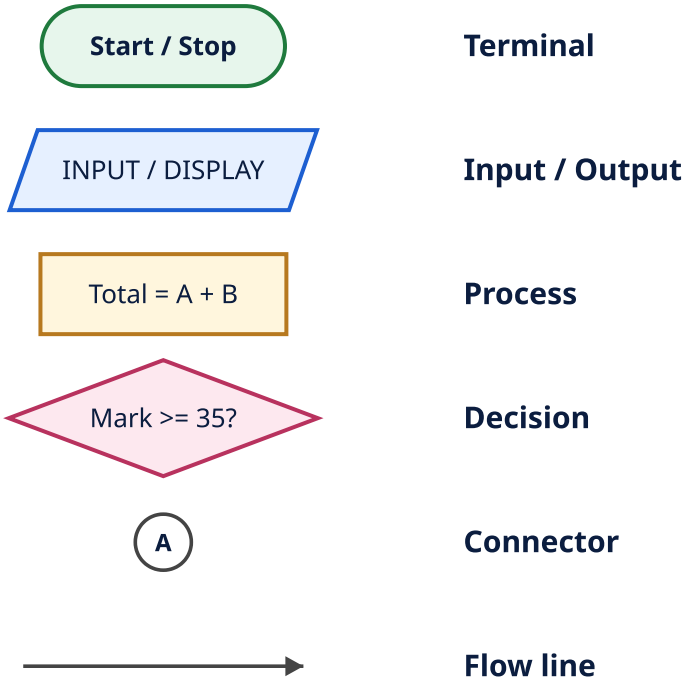
© 2026 schoolict.net (Ruwan Mangala Suraweera). All rights reserved.

G.C.E. O/L Grade 11 ICT · Lesson 1

1.3 Flowcharts and Pseudocode

Syllabus 10.3. We show an algorithm in two main ways. A **flowchart** is a **picture** made of standard symbols joined by arrows. **Pseudocode** is the algorithm written in **short English-like lines**, close to a real program. In this note you will learn the O/L flowchart symbols, the pseudocode words, how to **change one into the other**, and how to check an algorithm with a **trace table**.

1. Flowchart symbols



Standard flowchart symbols used in Sri Lankan O/L papers

Symbol	Name	Used for
Rounded rectangle (oval)	Terminal	Start and Stop
Parallelogram	Input / Output	INPUT, DISPLAY, PRINT
Rectangle	Process	Calculations, Total = A + B
Diamond	Decision	A Yes/No question, e.g. Mark >= 35?
Small circle	Connector	Joining flow lines
Arrow	Flow line	Shows the direction of the steps

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

> 💡 Rules for a neat flowchart: one Start and one Stop; arrows go top to bottom; a decision diamond has exactly **two** exits labelled **Yes** and **No**; write short text inside each box.

2. Pseudocode words

Pseudocode is not a real language, so small differences are allowed. But in O/L answers we normally use these words, written in CAPITALS:

Word	Meaning	Flowchart symbol
BEGIN / END	Start and end of the algorithm	Terminal
INPUT / READ	Get data	Input/Output
DISPLAY / PRINT	Show a result	Input/Output
Total = A + B	Calculate and store	Process
IF … THEN … ELSE … ENDIF	Selection	Decision
FOR … ENDFOR, WHILE … ENDWHILE, REPEAT … UNTIL	Iteration	Decision + arrow going back

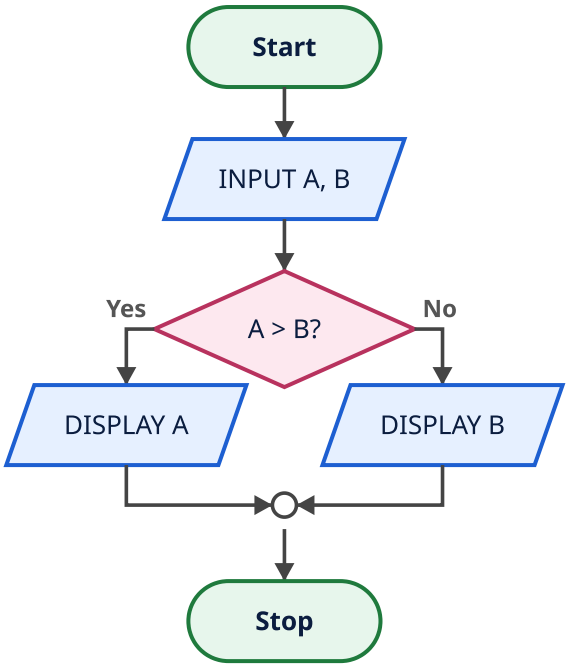
3. Example: the larger of two numbers

Look at the three forms below. Every flowchart box matches one pseudocode line, and every pseudocode line matches one or two lines of code.

Pseudocode

```
BEGIN
  INPUT A, B
  IF A > B THEN
    DISPLAY A
  ELSE
    DISPLAY B
  ENDIF
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – Largest of two numbers

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Largest of two numbers

Pascal

```
program Largest2;
var
  a, b: integer;
begin
  write('Enter first number: ');
  readln(a);
  write('Enter second number: ');
  readln(b);
  if a > b then
    writeln('Largest = ', a)
  else
    writeln('Largest = ', b);
end.
```

Python

```
a = int(input("Enter first number: "))
b = int(input("Enter second number: "))
if a > b:
    print("Largest =", a)
else:
    print("Largest =", b)
```

Sample run 1

```
Enter first number: 45
Enter second number: 72
Largest = 72
```

Sample run 2

```
Enter first number: 90
Enter second number: 34
Largest = 90
```

4. Converting a flowchart into pseudocode

Terminal → BEGIN / END
ආරම්භ/අවසාන → BEGIN / END

Parallelogram → INPUT / DISPLAY
සමාන්තරාස්‍රය → INPUT / DISPLAY

Rectangle → assignment
සෘජුකෝණාස්‍රය → පැවරීම

Diamond, no arrow back → IF
ආපසු ඊතලයක් නැති දියමන්තිය → IF

Diamond with an arrow back → loop
ආපසු ඊතලයක් ඇති දියමන්තිය → ලූපය

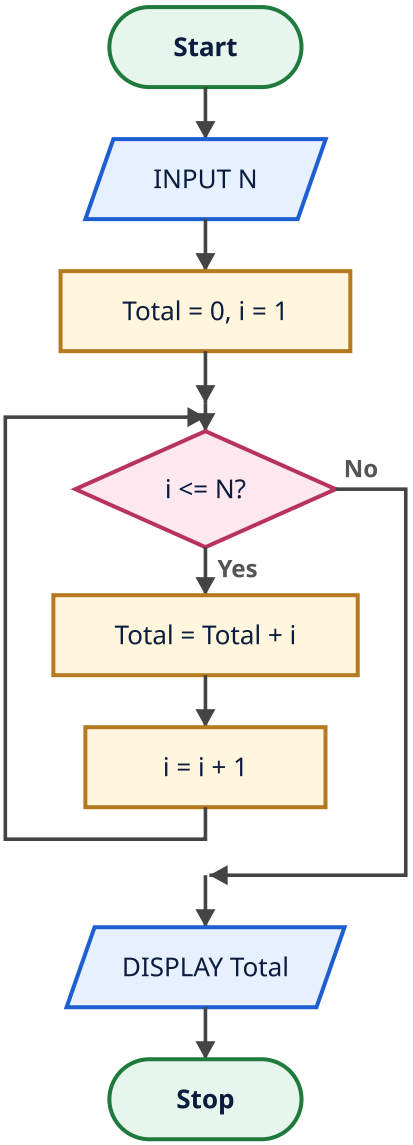
Converting **pseudocode into a flowchart** is the same table read backwards: draw one symbol for each line, and draw an arrow back to the diamond for every loop.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

5. Example with a loop: sum of 1 to N

Pseudocode

```
BEGIN
  INPUT N
  Total = 0
  i = 1
  WHILE i <= N DO
    Total = Total + i
    i = i + 1
  ENDWHILE
  DISPLAY Total
END
```



Flowchart – Sum of the numbers 1 to N

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Sum of the numbers 1 to N

Pascal

```
program SumToN;
var
  n, i, total: integer;
begin
  write('Enter N: ');
  readln(n);
  total := 0;
  i := 1;
  while i <= n do
  begin
    total := total + i;
    i := i + 1;
  end;
  writeln('Sum = ', total);
end.
```

Python

```
n = int(input("Enter N: "))
total = 0
i = 1
while i <= n:
    total = total + i
    i = i + 1
print("Sum =", total)
```

Sample run

```
Enter N: 5
Sum = 15
```

6. Trace tables (dry run)

A **trace table** shows the value of each variable after every step. Doing this on paper is called a **dry run**. It helps you find mistakes before running the program. Here is the trace of the program above for N = 5:

Step	i	Total	i <= 5 ?
start	1	0	–
loop	1	1	Yes
loop	2	3	Yes
loop	3	6	Yes
loop	4	10	Yes
loop	5	15	Yes
check	6	15	No

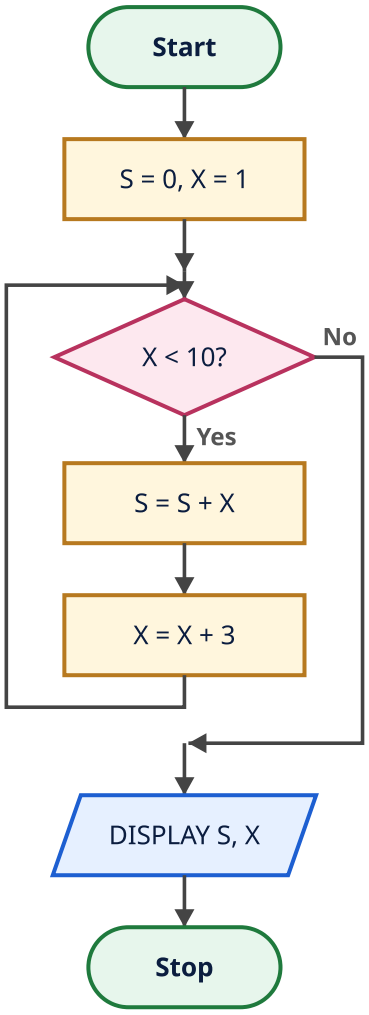
© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

The last row shows i = 6, so the condition i <= 5 is No and the loop stops. Total = **15**, which matches the real output “Sum = 15”.

7. Try it: trace this loop

Pseudocode

```
BEGIN
  S = 0
  X = 1
  WHILE X < 10 DO
    S = S + X
    X = X + 3
  ENDWHILE
  DISPLAY S, X
END
```



Flowchart – Trace this loop

Step	S	X	X < 10 ?
start	0	1	–
loop	1	4	Yes

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

loop	5	7	Yes
loop	12	10	Yes
check	12	10	No

Output: **12 10** (S = 12, X = 10). Check: 1 + 4 + 7 = 12.

Key Words

Flowchart – A diagram of an algorithm using standard symbols and arrows.

Pseudocode – An algorithm written in short English-like lines.

Terminal symbol – Rounded rectangle for Start and Stop.

Input/Output symbol – Parallelogram for INPUT and DISPLAY.

Process symbol – Rectangle for a calculation or assignment.

Decision symbol – Diamond for a Yes/No question.

Connector – Small circle that joins flow lines.

Flow line – Arrow showing the order of steps.

Trace table – A table showing variable values after each step.

Dry run – Following an algorithm by hand on paper.

Assignment – Storing a value in a variable, e.g. Total = 0.

Conversion – Changing a flowchart into pseudocode or the reverse.

Sample Q&A

1. Which symbol is used for a decision?

> **Answer:** A diamond with Yes and No exits.

2. Which symbol is used for INPUT and DISPLAY?

> **Answer:** The parallelogram.

3. What is pseudocode?

> **Answer:** An algorithm written in short English-like lines, close to a program.

4. How do you know a diamond is part of a loop?

> **Answer:** An arrow comes back up to it from below.

5. What is a trace table used for?

> **Answer:** To record variable values step by step and find mistakes (dry run).

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

6. Trace S = 0, X = 1; WHILE X < 10: S = S + X, X = X + 3. Give the final S and X.

> **Answer:** S = 12, X = 10.

MCQ

1. The flowchart symbol for Start and Stop is the…

A. rounded rectangle (terminal)
B. parallelogram
C. diamond
D. small circle

2. “Total = A + B” is drawn in a…

A. diamond
B. parallelogram
C. rectangle (process)
D. terminal

3. “DISPLAY Total” is drawn in a…

A. parallelogram
B. diamond
C. circle
D. rectangle

4. How many exits does a decision diamond have?

A. 3
B. 2
C. 1
D. 4

5. A small circle in a flowchart is a…

A. terminal
B. process
C. connector
D. decision

6. Pseudocode is…

A. a type of flowchart symbol
B. a programming language the computer runs
C. machine code
D. an algorithm in English-like lines

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

7. A diamond with an arrow coming back to it shows…

A. the end
B. a sequence
C. a loop
D. an input

8. Following an algorithm by hand on paper is called a…

A. upload
B. download
C. compile
D. dry run

9. In the sum 1 to N program with N = 5, the output is…

A. Sum = 5
B. Sum = 15
C. Sum = 10
D. Sum = 21

10. In that program, the value of i when the loop stops is…

A. 6
B. 15
C. 5
D. 0

11. For inputs 45 and 72, the larger-number program displays…

A. Largest = 45
B. Largest = 72
C. Largest = 27
D. Largest = 117

12. Trace: S = 0, X = 1; WHILE X < 10: S = S + X; X = X + 3. Output "S X" is…

A. 12 7
B. 22 13
C. 12 10
D. 10 12

13. When converting a flowchart, the Stop terminal becomes…

A. END
B. DISPLAY
C. ENDIF
D. INPUT

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

14. The Yes branch of a decision (no loop) goes after...

A. END
B. ENDIF
C. ELSE
D. THEN

15. Which is a rule for a good flowchart?

A. Three exits from each diamond
B. One Start and one Stop
C. Long paragraphs inside boxes
D. Arrows going in any direction with no labels

Structured Questions

1. Draw the flowchart symbols for (i) terminal (ii) input/output (iii) process (iv) decision and say what each is used for. (4 marks) *(4 marks)*

> Rounded rectangle – Start/Stop; parallelogram – INPUT/DISPLAY; rectangle – calculation/assignment; diamond – Yes/No question (1 each).

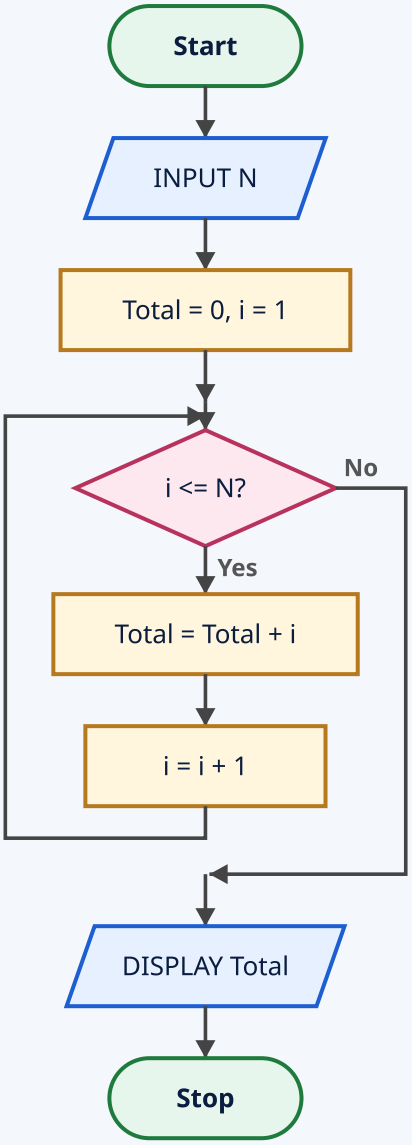
2. Write pseudocode to input two numbers and display the larger one. (4 marks) *(4 marks)*

> BEGIN; INPUT A, B; IF A > B THEN Largest = A ELSE Largest = B ENDIF; DISPLAY Largest; END (input 1, IF 2, display 1).
>
> **Pseudocode**
>
> ```
> BEGIN
>   INPUT A, B
>   IF A > B THEN
>     DISPLAY A
>   ELSE
>     DISPLAY B
>   ENDIF
> END
> ```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

3. Draw a flowchart to find the sum of the numbers 1 to N. (5 marks) *(5 marks)*

Start; input N; Total = 0, i = 1; diamond i <= N?; Yes → Total = Total + i, i = i + 1, arrow back; No → display Total; Stop (1 each for start/stop, input, initialise, loop with arrow back, display).



Flowchart – Sum of the numbers 1 to N

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

4. Complete a trace table for S = 0, X = 1; WHILE X < 10: S = S + X; X = X + 3; DISPLAY S, X. (4 marks) *(4 marks)*

S: 0, 1, 5, 12; X: 1, 4, 7, 10; condition Yes, Yes, Yes, No; output 12 10 (1 each).

Step	S	X	X < 10 ?
start	0	1	–
loop	1	4	Yes
loop	5	7	Yes
loop	12	10	Yes
check	12	10	No

5. Give two advantages of pseudocode over a flowchart, and one advantage of a flowchart. (3 marks) *(3 marks)*

Pseudocode: quicker to write and change; closer to real code (2). Flowchart: easy to see the flow as a picture (1).

Summary

- Flowchart = picture; pseudocode = English-like lines.
- Terminal (rounded) = Start/Stop.
- Parallelogram = input/output.
- Rectangle = process; diamond = decision.
- A diamond has Yes and No exits.
- Arrow back to a diamond = loop.
- Convert box by box, line by line.
- A trace table checks an algorithm (dry run).

MCQ answer key – 1.3

1. **A** – Terminal = rounded.
2. **C** – Calculation = process.
3. **A** – Output = parallelogram.
4. **B** – Yes and No.
5. **C** – Connector.
6. **D** – Not run by computer.
7. **C** – Arrow back = repeat.
8. **D** – Dry run.
9. **B** – 1+2+3+4+5 = 15.
10. **A** – Stops when i <= 5 is false.
11. **B** – 72 > 45.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

12. **C** – S=1,5,12; X=4,7,10.
13. **A** – Stop → END.
14. **D** – IF cond THEN yes ELSE no.
15. **B** – One Start, one Stop.

© 2026 schoolict.net (Ruwan Mangala Suraweera). All rights reserved.

G.C.E. O/L Grade 11 ICT · Lesson 1

1.4 Identifiers, Variables, Constants and Data Types

Syllabus 10.4. A program must remember data – a name, a mark, a price. It keeps each value in a **named box in memory**. If the value can change, the box is a **variable**; if it never changes, it is a **constant**. The name of the box is an **identifier**, and the kind of value it holds is its **data type**. Some words are **reserved** and cannot be used as names.

1. Variables

A **variable** is a named memory location whose value **can change** while the program runs. Think of a labelled box: the label is “Marks”, and inside today is 65; later we can put 80 in the same box. The old value is replaced.

Step	Pseudocode	Value in Marks
1	Marks = 65	65
2	Marks = 80	80
3	Marks = Marks + 5	85

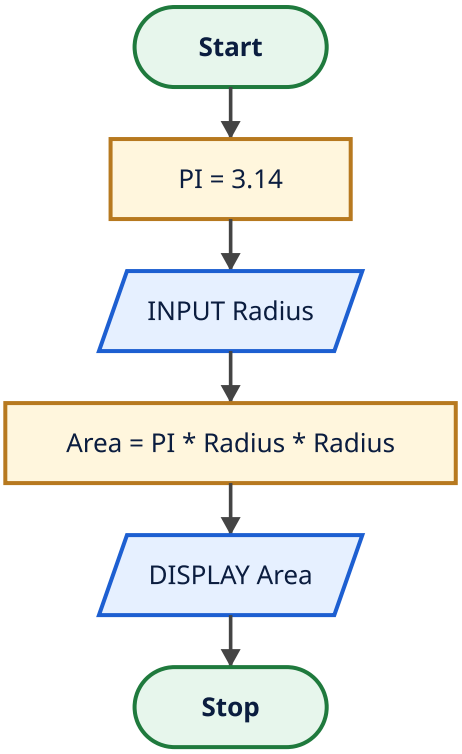
2. Constants

A **constant** is a named value that **does not change** while the program runs, such as π = 3.14 or the pass mark 35. Using a constant makes the program **easy to read** and **easy to change** – if the pass mark changes, you change it in one place only. In Pascal we write it under **const**. Python has no real constants, so by custom we write the name in CAPITALS.

Pseudocode

```
BEGIN
  CONSTANT PI = 3.14
  INPUT Radius
  Area = PI * Radius * Radius
  DISPLAY Area
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – A constant: area of a circle

A constant: area of a circle

Pascal

```
program CircleArea;
const
  PI = 3.14;
var
  radius, area: real;
begin
  write('Enter radius: ');
  readln(radius);
  area := PI * radius * radius;
  writeln('Area = ', area:0:2);
end.
```

Python

```
PI = 3.14            # a constant (capital letters by custom)
radius = float(input("Enter radius: "))
area = PI * radius * radius
print(f"Area = {area:.2f}")
```

Sample run

```
Enter radius: 7
Area = 153.86
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

3. Data types

The **data type** tells the computer what kind of value a variable holds and how much memory it needs. The O/L syllabus uses four basic types:

Data type	Holds	Examples	Pascal	Python
Integer	Whole numbers	16, −5, 2026	integer	int
Floating point (real)	Numbers with a decimal point	1.62, 3.14, 71.67	real	float
Character	One letter, digit or symbol	'A', '7', '#'	char	str of length 1
Logical (Boolean)	Only True or False	True, False	boolean	bool
String (extra)	Text – many characters	'Nimal', 'Kandy'	string	str

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Variables of different data types

Pascal

```
program DataTypes;
var
  name: string;
  age: integer;
  height: real;
  grade: char;
  isPrefect: boolean;
begin
  name := 'Nimal';
  age := 16;
  height := 1.62;
  grade := 'A';
  isPrefect := true;
  writeln('Name: ', name);
  writeln('Age: ', age);
  writeln('Height: ', height:0:2);
  writeln('Grade: ', grade);
  if isPrefect then
    writeln('Prefect: Yes')
  else
    writeln('Prefect: No');
end.
```

Python

```
name = "Nimal"        # string
age = 16              # integer
height = 1.62         # floating point (real)
grade = "A"           # character
is_prefect = True     # logical (Boolean)
print("Name:", name)
print("Age:", age)
print(f"Height: {height:.2f}")
print("Grade:", grade)
if is_prefect:
    print("Prefect: Yes")
else:
    print("Prefect: No")
```

Sample run

```
Name: Nimal
Age: 16
Height: 1.62
Grade: A
Prefect: Yes
```

💡 In **Pascal** you must **declare** every variable with its type before using it (var age: integer;). In **Python** you just assign a value (age = 16) and Python works out the type itself. Both are fine – but the idea of a data type is the same.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

4. Identifiers and naming rules

An **identifier** is the name we give to a variable, constant, program or sub-program. Rules (Pascal and Python):

- Use letters, digits and the underscore _ only.
- It must **start with a letter** (or _), never a digit.
- **No spaces** and no symbols such as -, %, @.
- It must **not be a reserved word**.
- Python is **case-sensitive** (Total and total are different). Pascal is **not** (Total and TOTAL are the same name).

Identifier	Valid?	Reason
total_marks	Yes	Letters and _
Mark2	Yes	Digit not at the start
2ndMark	No	Starts with a digit
total marks	No	Has a space
pass%	No	Has a symbol
begin / if	No	Reserved word

5. Reserved words (keywords)

Reserved words are words that already have a special meaning in the language, so they cannot be used as identifiers. Pascal: program, var, const, begin, end, if, then, else, for, while, repeat, until, and, or, not, div, mod. Python: if, else, elif, for, while, def, return, True, False, and, or, not, import.

6. Meaningful names

Choose names that **tell you what the value is**. Compare x = a * b with total = price * quantity. The second is much easier to understand and to fix later. Good: studentName, totalMarks, unit_price. Poor: a, x1, abc.

> 💡 Pascal trap: because Pascal ignores capital letters, a program called **Discount** cannot also have a variable called **discount** – the compiler says “Duplicate identifier”. Our discount program in 1.6 is therefore named **BillDiscount**.

Key Words

Variable – A named memory location whose value can change.

Constant – A named value that does not change while the program runs.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Identifier – The name of a variable, constant, program or sub-program.

Reserved word – A word with a special meaning in the language; not allowed as a name.

Data type – The kind of value a variable holds.

Integer – A whole number such as 16 or −5.

Floating point – A number with a decimal point such as 1.62 (Pascal: real).

Character – A single letter, digit or symbol such as 'A'.

Logical (Boolean) – A type with only two values: True and False.

Declaration – Telling the computer a variable's name and type before use.

Case-sensitive – Capital and small letters are treated as different (Python).

Meaningful name – A name that shows what the value is, e.g. totalMarks.

Sample Q&A

1. What is the difference between a variable and a constant?

> **Answer:** A variable's value can change while the program runs; a constant's value cannot.

2. Name the four basic data types in the syllabus.

> **Answer:** Integer, floating point (real), character and logical (Boolean).

3. Is 2ndMark a valid identifier? Why?

> **Answer:** No – an identifier cannot start with a digit.

4. Give the data type for: 16, 1.62, 'A', True.

> **Answer:** Integer, floating point, character, logical.

5. Why should we use meaningful names?

> **Answer:** The program is easier to read, understand and fix.

6. What is a reserved word? Give two examples.

> **Answer:** A word with a special meaning in the language, e.g. if, while (begin, end, for...).

MCQ

1. A named memory location whose value can change is a...

A. variable
B. reserved word
C. comment
D. constant

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

2. The value 3.14 used for π, which never changes, is best stored as a…

A. character
B. constant
C. variable
D. Boolean

3. Which is a valid identifier?

A. 2total
B. total marks
C. total_marks
D. total-marks

4. Which is NOT a valid identifier?

A. Mark2
B. unit_price
C. begin
D. studentName

5. The data type of 1.62 is…

A. integer
B. character
C. logical
D. floating point (real)

6. The data type of 'A' is…

A. integer
B. real
C. character
D. Boolean

7. A logical (Boolean) variable can hold…

A. only decimals
B. any text
C. any whole number
D. only True or False

8. The number of students in a class is best stored as…

A. integer
B. real
C. Boolean
D. character

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

9. In Pascal, the type for a floating point number is...

A. char
B. real
C. int
D. float

10. In Python, the type for a whole number is...

A. int
B. real
C. integer
D. bool

11. Which language is case-sensitive (Total ≠ total)?

A. Pascal
B. Both
C. Neither
D. Python

12. Which is the most meaningful name for the price of one item?

A. abc
B. a1
C. unitPrice
D. x

13. After Marks = 65, Marks = 80, Marks = Marks + 5, the value of Marks is...

A. 85
B. 150
C. 65
D. 70

14. In Pascal, variables are declared under the word...

A. const
B. var
C. def
D. begin

15. For radius 7 and PI = 3.14, the circle program shows...

A. Area = 21.98
B. Area = 153.86
C. Area = 43.96
D. Area = 49.00

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Structured Questions

1. (i) Define a variable and a constant. (2 marks)
(ii) Give one example of each from a school-fees program. (2 marks) *(4 marks)*

> (i) Variable – named memory location whose value can change; constant – named value that does not change. (ii) Variable: number of students / amount paid; constant: fee per term (e.g. 500).

2. Give the most suitable data type for: (a) a student's age (b) height in metres (c) a grade letter (d) whether a student is a prefect. (4 marks) *(4 marks)*

> (a) integer (b) floating point / real (c) character (d) logical / Boolean.

3. State whether each identifier is valid and give a reason: total, 3sum, my mark, if, Avg_1. (5 marks) *(5 marks)*

> total – valid; 3sum – invalid, starts with a digit; my mark – invalid, space; if – invalid, reserved word; Avg_1 – valid.

4. Explain two advantages of using a constant such as PASSMARK = 35 instead of typing 35 everywhere. (2 marks) *(2 marks)*

> Easy to read (the name tells the meaning) (1); easy to change – change in one place only (1).

5. A Pascal program called Discount has a variable discount. Why does it not compile? How can you fix it? (2 marks) *(2 marks)*

> Pascal is not case-sensitive, so both names are the same identifier ("Duplicate identifier") (1). Rename the program, e.g. BillDiscount (1).

Summary

- Variable: value can change.
- Constant: value stays fixed.
- Identifier = a name; start with a letter, no spaces.
- Reserved words cannot be names.
- Integer, floating point, character, logical.
- Pascal: integer, real, char, boolean.
- Python: int, float, str, bool; case-sensitive.
- Use meaningful names like totalMarks.

MCQ answer key – 1.4

1. **A** – Variable.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

2. **B** – Fixed value = constant.
3. **C** – Letters and underscore only.
4. **C** – begin is reserved in Pascal.
5. **D** – Has a decimal point.
6. **C** – Single letter.
7. **D** – Two values only.
8. **A** – Whole number.
9. **B** – Pascal uses real.
10. **A** – Python uses int.
11. **D** – Python is case-sensitive; Pascal is not.
12. **C** – Tells what it is.
13. **A** – 80 + 5.
14. **B** – var section.
15. **B** – 3.14 × 7 × 7 = 153.86.

© 2026 schoolict.net (Ruwan Mangala Suraweera). All rights reserved.

G.C.E. O/L Grade 11 ICT · Lesson 1

1.5 Operators: Arithmetic, Comparison and Logical

Syllabus 10.5. Operators are symbols that do work on values. **Arithmetic** operators calculate (+ − × ÷). **Comparison** (relational) operators compare two values and give True or False. **Logical** operators join conditions (AND, OR, NOT). When there are many operators in one line, **precedence** decides which is done first – just like BODMAS in maths.

1. Arithmetic operators

Operation	Pascal	Python	17 and 5
Addition	+	+	22
Subtraction	-	-	12
Multiplication	*	*	85
Division (decimal answer)	/	/	3.40
Whole-number division (quotient)	div	//	3
Remainder (modulus)	mod	%	2

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Arithmetic operators

Pascal

```
program Operators;
var
  a, b: integer;
begin
  a := 17;
  b := 5;
  writeln('Sum = ', a + b);
  writeln('Difference = ', a - b);
  writeln('Product = ', a * b);
  writeln('Division = ', a / b:0:2);
  writeln('Quotient = ', a div b);
  writeln('Remainder = ', a mod b);
end.
```

Python

```
a = 17
b = 5
print("Sum =", a + b)
print("Difference =", a - b)
print("Product =", a * b)
print(f"Division = {a / b:.2f}")
print("Quotient =", a // b)
print("Remainder =", a % b)
```

Sample run

```
Sum = 22
Difference = 12
Product = 85
Division = 3.40
Quotient = 3
Remainder = 2
```

💡 **mod** is very useful: N mod 2 = 0 means N is **even**. 17 mod 5 = 2 because 17 = 3 × 5 + 2.

2. Comparison operators

A comparison always gives a **logical** result: True or False. Be careful – Pascal and Python write “equal” and “not equal” differently.

Meaning	Pascal / pseudocode	Python	Example (A = 8, B = 5)
Equal to	=	==	A = B → False
Not equal to	<>	!=	A <> B → True
Greater than	>	>	A > B → True
Less than	<	<	A < B → False
Greater than or equal	>=	>=	A >= 8 → True

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Less than or equal	<=	<=	B <= 4 → False

3. Logical operators

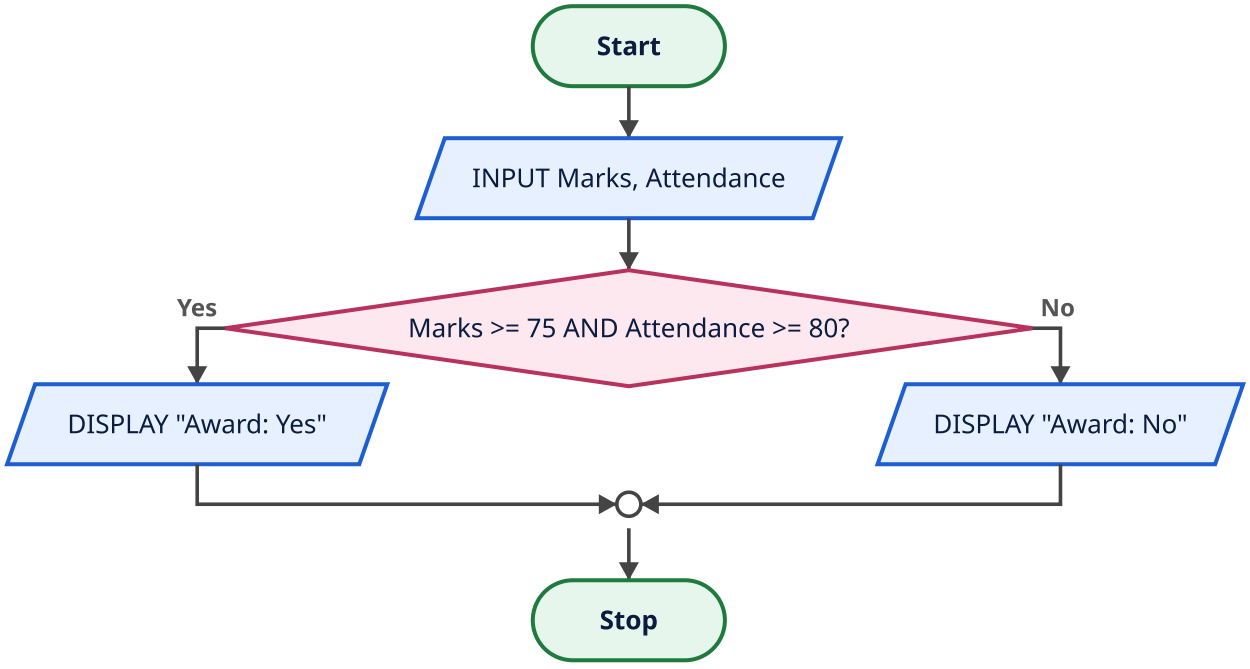
AND is True only when **both** conditions are True. **OR** is True when **at least one** is True. **NOT** reverses the value. (You met these as logic gates in Grade 10.)

A	B	A AND B	A OR B	NOT A
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Example: a scholarship is given only if marks ≥ 75 **AND** attendance ≥ 80%.

Pseudocode

```
BEGIN
  INPUT Marks, Attendance
  IF Marks >= 75 AND Attendance >= 80 THEN
    DISPLAY "Award: Yes"
  ELSE
    DISPLAY "Award: No"
  ENDIF
END
```



Flowchart – Logical operator AND: scholarship award

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Logical operator AND: scholarship award

Pascal

```
program Award;
var
  marks, attendance: integer;
begin
  write('Enter marks: ');
  readln(marks);
  write('Enter attendance %: ');
  readln(attendance);
  if (marks >= 75) and (attendance >= 80) then
    writeln('Award: Yes')
  else
    writeln('Award: No');
end.
```

Python

```
marks = int(input("Enter marks: "))
attendance = int(input("Enter attendance %: "))
if marks >= 75 and attendance >= 80:
    print("Award: Yes")
else:
    print("Award: No")
```

Sample run 1

```
Enter marks: 80
Enter attendance %: 85
Award: Yes
```

Sample run 2

```
Enter marks: 80
Enter attendance %: 70
Award: No
```

4. Operator precedence

Order	Operators
1	( ) brackets
2	NOT
3	* / div mod (Python: * / // %)
4	+ −
5	Comparison = <> < > <= >=
6	AND, then OR

Operators at the **same level** are done **left to right**. So 20 − 8 / 4 × 2 = 20 − 2 × 2 = 20 − 4 = 16, and 10 − 4 − 3 = 6 − 3 = 3. When in doubt, **use brackets** – they make the order clear to everyone.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Operator precedence

Pascal

```
program Precedence;
begin
  writeln('2 + 3 * 4 = ', 2 + 3 * 4);
  writeln('(2 + 3) * 4 = ', (2 + 3) * 4);
  writeln('20 - 8 / 4 * 2 = ', 20 - 8 / 4 * 2:0:2);
  writeln('10 - 4 - 3 = ', 10 - 4 - 3);
end.
```

Python

```
print("2 + 3 * 4 =", 2 + 3 * 4)
print("(2 + 3) * 4 =", (2 + 3) * 4)
print("20 - 8 / 4 * 2 =", f"{20 - 8 / 4 * 2:.2f}")
print("10 - 4 - 3 =", 10 - 4 - 3)
```

Sample run

```
2 + 3 * 4 = 14
(2 + 3) * 4 = 20
20 - 8 / 4 * 2 = 16.00
10 - 4 - 3 = 3
```

💡 Pascal note: in Pascal, AND and OR are done **before** comparisons, so you must put brackets: if (Marks >= 75) and (Att >= 80) then ... Python does not need them, but brackets never hurt.

Key Words

Operator – A symbol that does work on values, e.g. +.

Operand – A value an operator works on, e.g. 17 and 5 in 17 + 5.

Arithmetic operator – Calculates: + − * / div mod.

div / // – Whole-number division: 17 div 5 = 3.

mod / % – Remainder after division: 17 mod 5 = 2.

Comparison operator – Compares two values and gives True or False.

Not equal – <> in Pascal, != in Python.

Logical operator – Joins or reverses conditions: AND, OR, NOT.

AND – True only if both conditions are True.

OR – True if at least one condition is True.

NOT – Changes True to False and False to True.

Precedence – The order in which operators are done.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Sample Q&A

1. Find 17 div 5 and 17 mod 5.

> **Answer:** 3 and 2.

2. How do you write “not equal” in Pascal and Python?

> **Answer:** Pascal <>, Python !=.

3. When is A AND B True?

> **Answer:** Only when both A and B are True.

4. Evaluate 2 + 3 * 4.

> **Answer:** 14 (multiply first).

5. Evaluate 20 − 8 / 4 * 2.

> **Answer:** 16 (8 / 4 = 2, 2 × 2 = 4, 20 − 4 = 16).

6. How can you test whether N is even?

> **Answer:** N mod 2 = 0 (Python: N % 2 == 0).

MCQ

1. 17 mod 5 is...

A. 12
B. 3
C. 3.4
D. 2

2. 17 div 5 is...

A. 85
B. 3.4
C. 3
D. 2

3. In Python, whole-number division is written...

A. //
B. /
C. div
D. %

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

4. In Python, the remainder operator is…

A. #
B. %
C. //
D. mod

5. In Pascal, “not equal to” is written…

A. <>
B. ==
C. !=
D. =<

6. In Python, “equal to” in a comparison is written…

A. <>
B. :=
C. ==
D. =

7. A comparison operator always gives…

A. a decimal
B. a string
C. True or False
D. a character

8. True AND False gives…

A. Error
B. 1
C. True
D. False

9. True OR False gives…

A. False
B. 0
C. True
D. Error

10. NOT True gives…

A. False
B. 0.5
C. Nothing
D. True

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

11. 2 + 3 * 4 =

A. 9
B. 14
C. 24
D. 20

12. (2 + 3) * 4 =

A. 9
B. 20
C. 24
D. 14

13. 10 − 4 − 3 =

A. 3
B. 9
C. 17
D. −3

14. Marks = 80, attendance = 70. “Marks >= 75 AND attendance >= 80” is…

A. True
B. 80
C. Error
D. False

15. Which operator is done first?

A. +
B. ( ) brackets
C. AND
D. >=

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Structured Questions

1. Using A = 17 and B = 5, find: (a) A + B (b) A / B (c) A div B (d) A mod B. (4 marks) *(4 marks)*

(a) 22 (b) 3.4 (c) 3 (d) 2.

Arithmetic operators

Pascal

```
program Operators;
var
  a, b: integer;
begin
  a := 17;
  b := 5;
  writeln('Sum = ', a + b);
  writeln('Difference = ', a - b);
  writeln('Product = ', a * b);
  writeln('Division = ', a / b:0:2);
  writeln('Quotient = ', a div b);
  writeln('Remainder = ', a mod b);
end.
```

Python

```
a = 17
b = 5
print("Sum =", a + b)
print("Difference =", a - b)
print("Product =", a * b)
print(f"Division = {a / b:.2f}")
print("Quotient =", a // b)
print("Remainder =", a % b)
```

Sample run

```
Sum = 22
Difference = 12
Product = 85
Division = 3.40
Quotient = 3
Remainder = 2
```

2. Complete the truth table for A AND B, A OR B and NOT A. (4 marks) *(4 marks)*

AND: T only for T,T; OR: F only for F,F; NOT A: reverse of A (see the table in the notes).

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

3. Evaluate, showing steps: (a) 2 + 3 * 4 (b) (2 + 3) * 4 (c) 20 − 8 / 4 * 2. (3 marks) *(3 marks)*

(a) 2 + 12 = 14 (b) 5 × 4 = 20 (c) 20 − 2 × 2 = 20 − 4 = 16.

Operator precedence

Pascal

```
program Precedence;
begin
  writeln('2 + 3 * 4 = ', 2 + 3 * 4);
  writeln('(2 + 3) * 4 = ', (2 + 3) * 4);
  writeln('20 - 8 / 4 * 2 = ', 20 - 8 / 4 * 2:0:2);
  writeln('10 - 4 - 3 = ', 10 - 4 - 3);
end.
```

Python

```
print("2 + 3 * 4 =", 2 + 3 * 4)
print("(2 + 3) * 4 =", (2 + 3) * 4)
print("20 - 8 / 4 * 2 =", f"{20 - 8 / 4 * 2:.2f}")
print("10 - 4 - 3 =", 10 - 4 - 3)
```

Sample run

```
2 + 3 * 4 = 14
(2 + 3) * 4 = 20
20 - 8 / 4 * 2 = 16.00
10 - 4 - 3 = 3
```

4. A student gets a scholarship if marks ≥ 75 and attendance ≥ 80. Write the condition, and give the result for marks 80 / attendance 85 and marks 80 / attendance 70. (3 marks) *(3 marks)*

(Marks >= 75) AND (Attendance >= 80) (1); 80/85 → Yes (1); 80/70 → No (1).

5. Explain the difference between = and == in Python. (2 marks) *(2 marks)*

= assigns (stores) a value in a variable (1); == compares two values and gives True or False (1).

Summary

- Arithmetic: + − * / div mod.
- Python uses // for div and % for mod.
- Comparison gives True or False.
- Not equal: Pascal <>, Python !=.
- AND: both true; OR: one true; NOT: reverse.
- Brackets first, then * /, then + −.
- Same level: left to right.
- N mod 2 = 0 means N is even.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

MCQ answer key – 1.5

1. **D** – Remainder 2.
2. **C** – Whole quotient.
3. **A** – //.
4. **B** – %.
5. **A** – <>.
6. **C** – == compares; = assigns.
7. **C** – Logical result.
8. **D** – AND needs both true.
9. **C** – OR needs one true.
10. **A** – NOT reverses.
11. **B** – Multiply first.
12. **B** – Brackets first.
13. **A** – Left to right.
14. **D** – Second part false.
15. **B** – Brackets highest.

© 2026 schoolict.net (Ruwan Mangala Suraweera). All rights reserved.

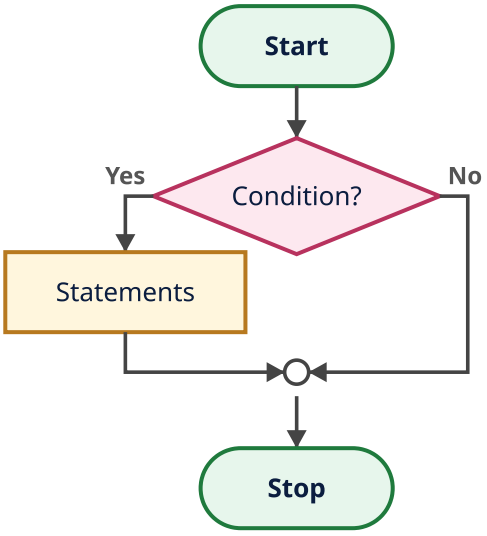
G.C.E. O/L Grade 11 ICT · Lesson 1

1.6 Selection: If–EndIf, If–Else–EndIf and Case

Syllabus 10.6. Selection lets a program make decisions. There are three forms: **If–EndIf** (do something only when a condition is true), **If–Else–EndIf** (choose between two paths) and **Case** (choose one of many options). For each one we go **flowchart → pseudocode → Pascal and Python code**.

1. If–EndIf (one-way selection)

If the condition is **True**, the statements inside run. If it is **False**, they are **skipped** and the program continues after ENDIF.



General form of If–EndIf

General pseudocode

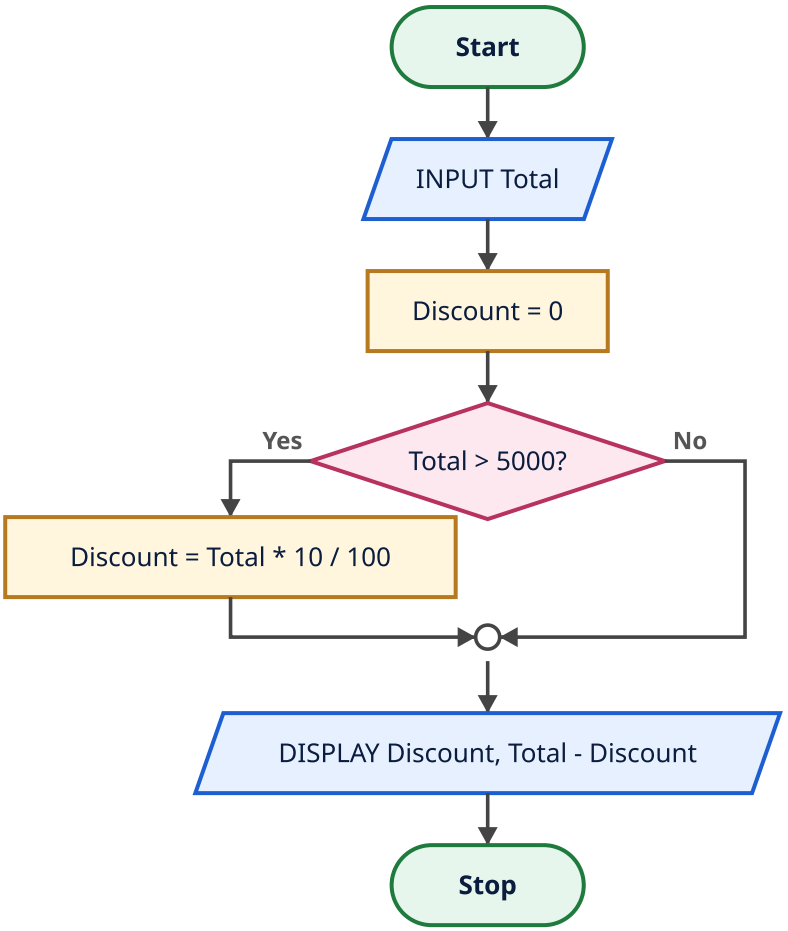
```
IF condition THEN
  statements
ENDIF
```

Example: a shop in Galle gives a **10% discount** when the bill is **more than Rs. 5000**.

Pseudocode

```
BEGIN
  INPUT Total
  Discount = 0
  IF Total > 5000 THEN
    Discount = Total * 10 / 100
  ENDIF
  DISPLAY Discount, Total - Discount
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – If–EndIf: 10% discount on a bill over Rs. 5000

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

If–EndIf: 10% discount on a bill over Rs. 5000

Pascal

```
program BillDiscount;
var
  total, discount: integer;
begin
  write('Enter bill total (Rs.): ');
  readln(total);
  discount := 0;
  if total > 5000 then
    discount := total * 10 div 100;
  writeln('Discount = Rs. ', discount);
  writeln('Amount to pay = Rs. ', total - discount);
end.
```

Python

```
total = int(input("Enter bill total (Rs.): "))
discount = 0
if total > 5000:
    discount = total * 10 // 100
print("Discount = Rs.", discount)
print("Amount to pay = Rs.", total - discount)
```

Sample run 1

```
Enter bill total (Rs.): 6500
Discount = Rs. 650
Amount to pay = Rs. 5850
```

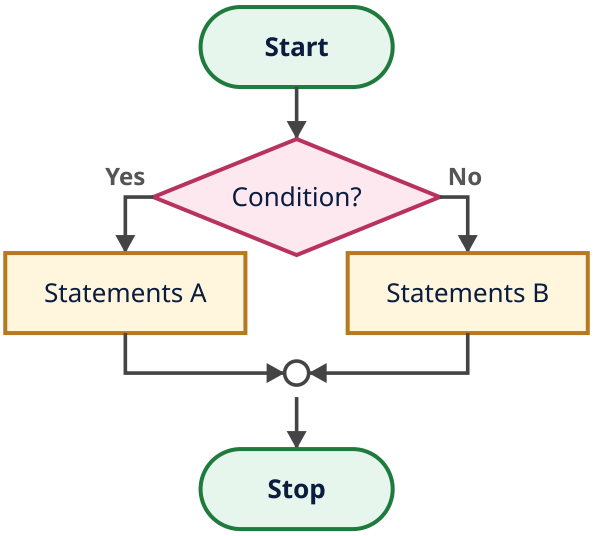
Sample run 2

```
Enter bill total (Rs.): 3000
Discount = Rs. 0
Amount to pay = Rs. 3000
```

2. If–Else–EndIf (two-way selection)

Exactly **one of two** groups of statements runs: the THEN part if the condition is True, the ELSE part if it is False.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



General form of If–Else–EndIf

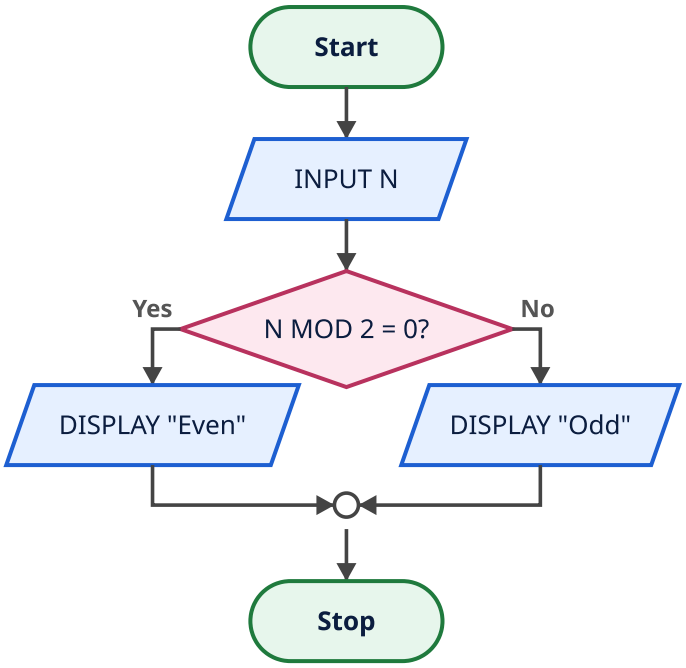
General pseudocode

```
IF condition THEN
  statements A
ELSE
  statements B
ENDIF
```

Pseudocode

```
BEGIN
  INPUT N
  IF N MOD 2 = 0 THEN
    DISPLAY "Even"
  ELSE
    DISPLAY "Odd"
  ENDIF
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – If–Else–EndIf: even or odd

If–Else–EndIf: even or odd

Pascal

```
program EvenOdd;
var
  n: integer;
begin
  write('Enter a number: ');
  readln(n);
  if n mod 2 = 0 then
    writeln(n, ' is even')
  else
    writeln(n, ' is odd');
end.
```

Python

```
n = int(input("Enter a number: "))
if n % 2 == 0:
    print(n, "is even")
else:
    print(n, "is odd")
```

Sample run 1

```
Enter a number: 14
14 is even
```

Sample run 2

```
Enter a number: 7
7 is odd
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

> 💡 Pascal trap: **never put a semicolon before else**. Write writeln('even') else … not writeln('even'); else … – the semicolon ends the whole if statement and the compiler gives an error. Python instead uses a colon : and **indentation** (4 spaces).

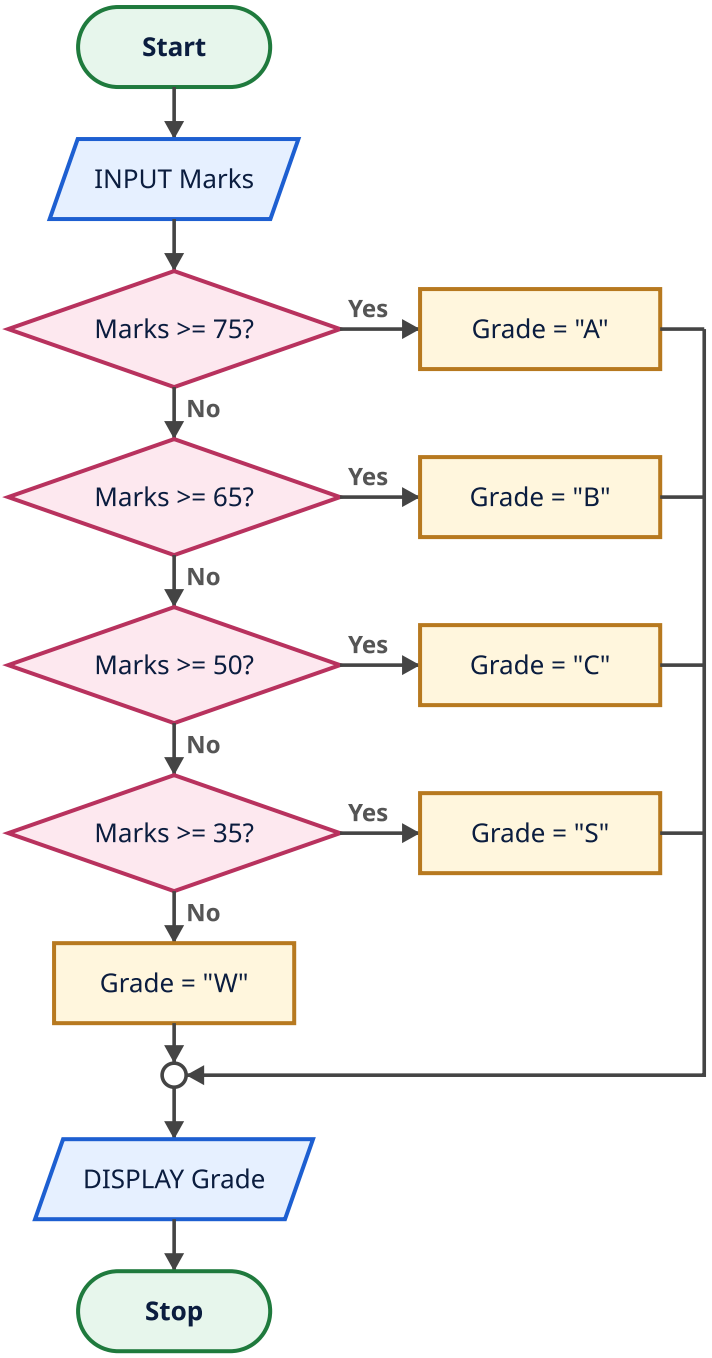
3. Many choices: IF … ELSE IF (O/L grades)

When there are more than two paths, we check conditions **one after another**. The first True condition wins and the rest are skipped. Sri Lankan O/L grades: A ≥ 75, B ≥ 65, C ≥ 50, S ≥ 35, otherwise W.

Pseudocode

```
BEGIN
  INPUT Marks
  IF Marks >= 75 THEN
    Grade = "A"
  ELSE IF Marks >= 65 THEN
    Grade = "B"
  ELSE IF Marks >= 50 THEN
    Grade = "C"
  ELSE IF Marks >= 35 THEN
    Grade = "S"
  ELSE
    Grade = "W"
  ENDIF
  DISPLAY Grade
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – Marks to grade (A, B, C, S, W)

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Marks to grade (A, B, C, S, W)

Pascal

```
program MarksToGrade;
var
  marks: integer;
  grade: char;
begin
  write('Enter marks: ');
  readln(marks);
  if marks >= 75 then
    grade := 'A'
  else if marks >= 65 then
    grade := 'B'
  else if marks >= 50 then
    grade := 'C'
  else if marks >= 35 then
    grade := 'S'
  else
    grade := 'W';
  writeln('Grade: ', grade);
end.
```

Python

```
marks = int(input("Enter marks: "))
if marks >= 75:
    grade = "A"
elif marks >= 65:
    grade = "B"
elif marks >= 50:
    grade = "C"
elif marks >= 35:
    grade = "S"
else:
    grade = "W"
print("Grade:", grade)
```

Sample run 1

```
Enter marks: 82
Grade: A
```

Sample run 2

```
Enter marks: 58
Grade: C
```

Sample run 3

```
Enter marks: 30
Grade: W
```

4. Case (multi-way selection)

When one variable is compared with many **fixed values** (1, 2, 3…), **Case** is shorter and clearer than many IFs. Pascal uses **case … of … else … end**. Python (3.10 and later) uses **match … case**; case _

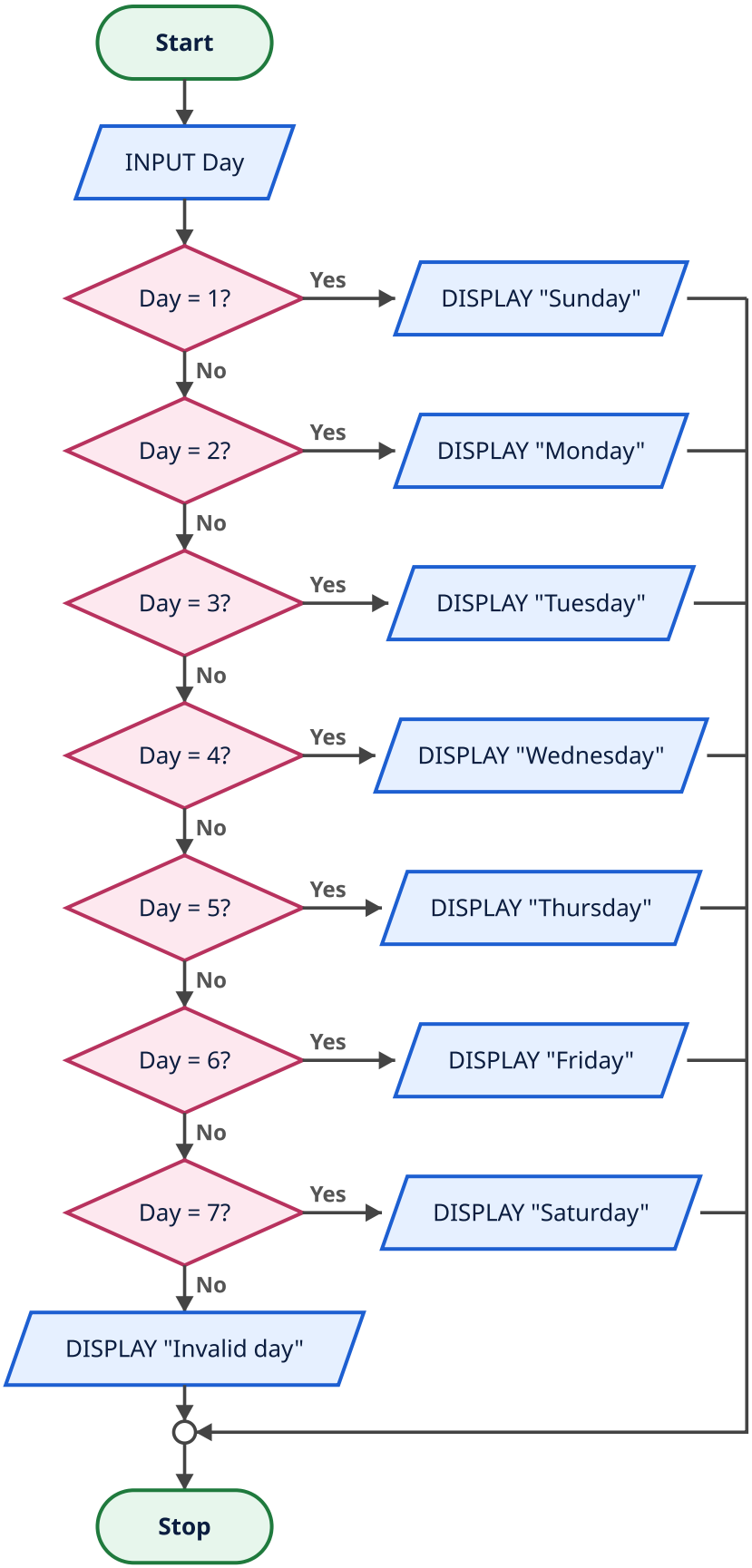
© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

means “anything else”.

Pseudocode

```
BEGIN
  INPUT Day
  CASE Day OF
    1 : DISPLAY "Sunday"
    2 : DISPLAY "Monday"
    3 : DISPLAY "Tuesday"
    4 : DISPLAY "Wednesday"
    5 : DISPLAY "Thursday"
    6 : DISPLAY "Friday"
    7 : DISPLAY "Saturday"
    OTHERWISE : DISPLAY "Invalid day"
  ENDCASE
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – Case / switch: day number to day name

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Case / switch: day number to day name

Pascal

```
program DayName;
var
  day: integer;
begin
  write('Enter day number (1-7): ');
  readln(day);
  case day of
    1: writeln('Sunday');
    2: writeln('Monday');
    3: writeln('Tuesday');
    4: writeln('Wednesday');
    5: writeln('Thursday');
    6: writeln('Friday');
    7: writeln('Saturday');
  else
    writeln('Invalid day');
  end;
end.
```

Python

```
day = int(input("Enter day number (1-7): "))
match day:
    case 1:
        print("Sunday")
    case 2:
        print("Monday")
    case 3:
        print("Tuesday")
    case 4:
        print("Wednesday")
    case 5:
        print("Thursday")
    case 6:
        print("Friday")
    case 7:
        print("Saturday")
    case _:
        print("Invalid day")
```

Sample run 1

```
Enter day number (1-7): 3
Tuesday
```

Sample run 2

```
Enter day number (1-7): 9
Invalid day
```

5. From flowchart to code – the steps

1. Read the diamond
1. දියමන්තිය කියවන්න

2. Yes branch
2. ඔව් ශාඛාව

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

3. No branch
3. නැත ශාඛාව

4. Where they join
4. ඒවා එක් වන තැන

5. Test both paths
5. මාර්ග දෙකම පරීක්ෂා කරන්න

Key Words

Selection – Choosing which statements run using a condition.

If–EndIf – Runs statements only when the condition is True.

If–Else–EndIf – Chooses one of two groups of statements.

Case – Chooses one of many options using fixed values.

THEN – Starts the part that runs when the condition is True.

ELSE – Starts the part that runs when the condition is False.

ENDIF – Marks the end of an IF in pseudocode.

match … case – Python's case statement (Python 3.10+).

Indentation – Spaces at the start of a line; Python uses it to group statements.

Branch – One of the paths out of a decision.

Even number – A number with N mod 2 = 0.

Test data – Inputs chosen to check every path of a program.

Sample Q&A

1. What is the difference between If–EndIf and If–Else–EndIf?

> **Answer:** If–EndIf does something or nothing; If–Else–EndIf always does one of two things.

2. For a bill of Rs. 6500, what discount does the program give?

> **Answer:** Rs. 650; amount to pay Rs. 5850.

3. What grade does 58 get?

> **Answer:** C (≥ 50 but < 65).

4. When is Case better than many IFs?

> **Answer:** When one variable is compared with many fixed values.

5. What does the day program display for 9?

> **Answer:** Invalid day.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

6. Why must we test both paths of an IF?

> **Answer:** A mistake may be hidden in the path we did not test.

MCQ

1. In If–EndIf, when the condition is False…

A. an error occurs
B. the program stops
C. the statements run twice
D. the statements are skipped

2. If–Else–EndIf always runs…

A. the ELSE part only
B. exactly one of two parts
C. both parts
D. neither part

3. Bill Rs. 6500 with 10% discount over Rs. 5000. Amount to pay is…

A. Rs. 650
B. Rs. 5000
C. Rs. 5850
D. Rs. 6500

4. Bill Rs. 3000 in the same program. Discount is…

A. Rs. 0
B. Rs. 3000
C. Rs. 300
D. Rs. 30

5. The even/odd program shows “7 is odd” because…

A. 7 > 2
B. 7 mod 2 = 1
C. 7 mod 2 = 0
D. 7 div 2 = 3

6. The O/L grade for 82 marks is…

A. S
B. C
C. A
D. B

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

7. The O/L grade for 30 marks is…

A. W
B. C
C. F
D. S

8. The O/L grade for 66 marks is…

A. A
B. B
C. C
D. S

9. Choosing a day name from a day number 1–7 is best done with…

A. Case
B. an array of Booleans
C. a FOR loop
D. a constant

10. Python’s case statement uses the words…

A. select … when
B. switch … case
C. match … case
D. case … of

11. In Pascal, what must NOT come just before else?

A. a new line
B. a space
C. writeln
D. a semicolon ;

12. Python groups the statements inside an if by…

A. brackets { }
B. semicolons
C. indentation
D. begin … end

13. In a flowchart, the Yes branch of a decision becomes the… part.

A. ELSE
B. THEN
C. CASE
D. ENDIF

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

14. In Pascal case, the part for “any other value” is written…

A. otherwise only in Python
B. default
C. case _
D. else

15. To test a pass/fail program fully, use…

A. one passing and one failing mark
B. the same mark twice
C. only a passing mark
D. no input

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

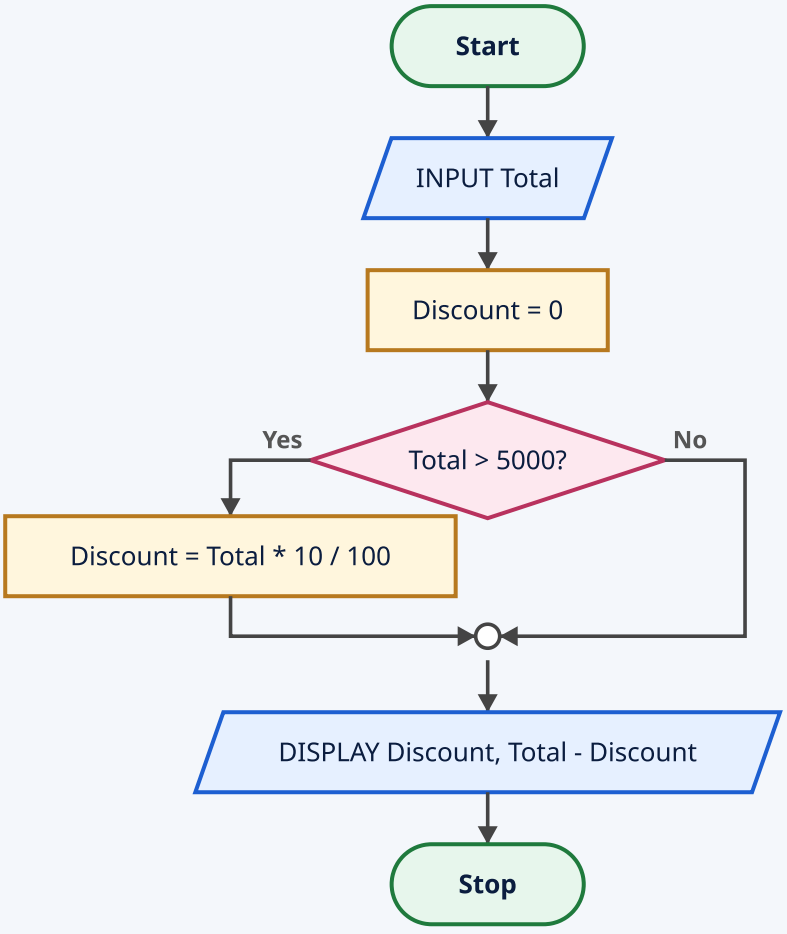
Structured Questions

1. A shop gives a 10% discount when the bill is more than Rs. 5000. (i) Draw the flowchart. (ii) Write the pseudocode. (iii) Find the amount to pay for Rs. 6500 and Rs. 3000. (6 marks) *(6 marks)*

(i) Start, input Total, Discount = 0, diamond Total > 5000?, Yes → Discount = Total × 10 / 100, display, Stop (2). (ii) As shown (2). (iii) Rs. 5850 and Rs. 3000 (2).

Pseudocode

```
BEGIN
  INPUT Total
  Discount = 0
  IF Total > 5000 THEN
    Discount = Total * 10 / 100
  ENDIF
  DISPLAY Discount, Total - Discount
END
```



Flowchart – If–EndIf: 10% discount on a bill over Rs. 5000

2. Write a program (Pascal or Python) to input a number and display whether it is even or odd. (4 marks) *(4 marks)*

Input (1); condition N mod 2 = 0 (1); display even (1); else display odd (1).

If–Else–EndIf: even or odd

Pascal

```
program EvenOdd;
var
  n: integer;
begin
  write('Enter a number: ');
  readln(n);
  if n mod 2 = 0 then
    writeln(n, ' is even')
  else
    writeln(n, ' is odd');
end.
```

Python

```
n = int(input("Enter a number: "))
if n % 2 == 0:
    print(n, "is even")
else:
    print(n, "is odd")
```

Sample run 1

```
Enter a number: 14
14 is even
```

Sample run 2

```
Enter a number: 7
7 is odd
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

3. Write pseudocode that inputs marks and displays the O/L grade (A ≥ 75, B ≥ 65, C ≥ 50, S ≥ 35, W otherwise). (5 marks) *(5 marks)*

> Input (1); IF ≥ 75 A (1); ELSE IF ≥ 65 B, ≥ 50 C (1); ≥ 35 S (1); ELSE W and display (1).
> **Pseudocode**

```
BEGIN
  INPUT Marks
  IF Marks >= 75 THEN
    Grade = "A"
  ELSE IF Marks >= 65 THEN
    Grade = "B"
  ELSE IF Marks >= 50 THEN
    Grade = "C"
  ELSE IF Marks >= 35 THEN
    Grade = "S"
  ELSE
    Grade = "W"
  ENDIF
  DISPLAY Grade
END
```

4. (i) When is a Case statement used? (1 mark)
(ii) Write a Case statement in pseudocode that displays “Sunday” for 1, “Monday” for 2 and “Invalid day” otherwise. (3 marks) *(4 marks)*

> (i) When one variable is compared with many fixed values. (ii) CASE Day OF 1 : DISPLAY “Sunday”; 2 : DISPLAY “Monday”; OTHERWISE : DISPLAY “Invalid day”; ENDCASE.

5. Find the error in this Pascal code and correct it: if Mark >= 35 then writeln('Pass'); else writeln('Fail'); (2 marks) *(2 marks)*

> There must be no semicolon before else (1). Correct: if Mark >= 35 then writeln('Pass') else writeln('Fail'); (1).

Summary

- Selection = making a decision.
- If–EndIf: do it or skip it.
- If–Else–EndIf: one of two paths.
- ELSE IF chains check conditions in order.
- Case: one of many fixed values.
- Pascal: case … of; Python: match … case.
- Pascal: no ; before else. Python: indent.
- Test every path with test data.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

MCQ answer key – 1.6

1. **D** – Skipped.
2. **B** – One of two.
3. **C** – 6500 − 650.
4. **A** – Not over 5000.
5. **B** – Remainder 1.
6. **C** – ≥ 75.
7. **A** – < 35.
8. **B** – 65–74.
9. **A** – Many fixed values.
10. **C** – Python 3.10+.
11. **D** – Semicolon ends the if.
12. **C** – Indentation.
13. **B** – Yes → THEN.
14. **D** – Pascal: else.
15. **A** – Test both paths.

© 2026 schoolict.net (Ruwan Mangala Suraweera). All rights reserved.

G.C.E. O/L Grade 11 ICT · Lesson 1

1.7 Iteration: FOR, WHILE and REPEAT–UNTIL Loops

Syllabus 10.7. A **loop** repeats statements. If we **know how many times** to repeat (e.g. 10 times), we use a **FOR** loop. If we **don't know** (e.g. "until the user types −1"), we use a **WHILE** loop, which **tests the condition at the beginning**, or a **REPEAT–UNTIL** loop, which **tests at the end**. You will also learn to **convert** one loop into another.

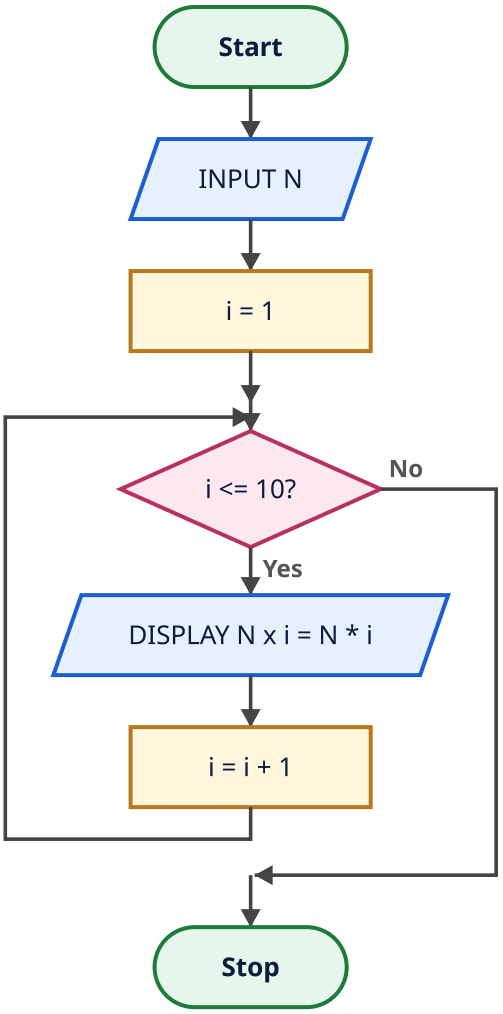
1. Known number of times: FOR

A **FOR** loop uses a **counter** that goes from a start value to an end value. The loop body runs once for each value. Example: the multiplication table of a number, 1 to 10.

Pseudocode

```
BEGIN
  INPUT N
  FOR i = 1 TO 10
    DISPLAY N, " x ", i, " = ", N * i
  ENDFOR
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – For loop (known count): multiplication table

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

For loop (known count): multiplication table

Pascal

```
program TimesTable;
var
  n, i: integer;
begin
  write('Enter a number: ');
  readln(n);
  for i := 1 to 10 do
    writeln(n, ' x ', i, ' = ', n * i);
end.
```

Python

```
n = int(input("Enter a number: "))
for i in range(1, 11):
    print(n, "x", i, "=", n * i)
```

Sample run

```
Enter a number: 7
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
```

Trace of a FOR loop – factorial of 5 (F = 1 × 2 × 3 × 4 × 5):

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Factorial with a FOR loop

Pascal

```
program Factorial;
var
  n, i, f: integer;
begin
  write('Enter N: ');
  readln(n);
  f := 1;
  for i := 1 to n do
    f := f * i;
  writeln('Factorial = ', f);
end.
```

Python

```
n = int(input("Enter N: "))
f = 1
for i in range(1, n + 1):
    f = f * i
print("Factorial =", f)
```

Sample run

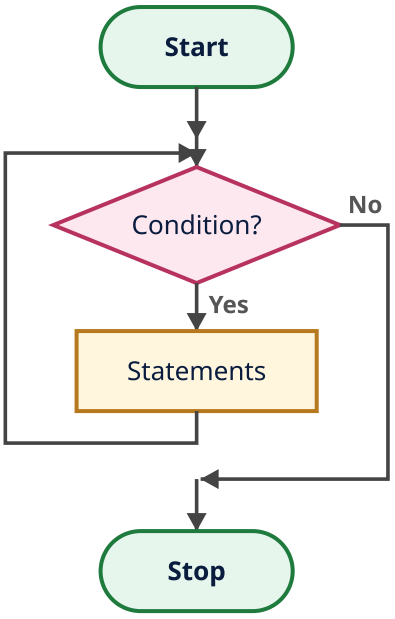
```
Enter N: 5
Factorial = 120
```

i	F = F * i
1	1
2	2
3	6
4	24
5	120

2. Unknown count, test at the beginning: WHILE

A **WHILE** loop checks the condition **first**. If it is True, the body runs and the condition is checked again. If it is False at the start, the body runs **zero times**. Example: enter marks until −1 is typed. −1 is a **sentinel** (a stop signal), not a mark.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

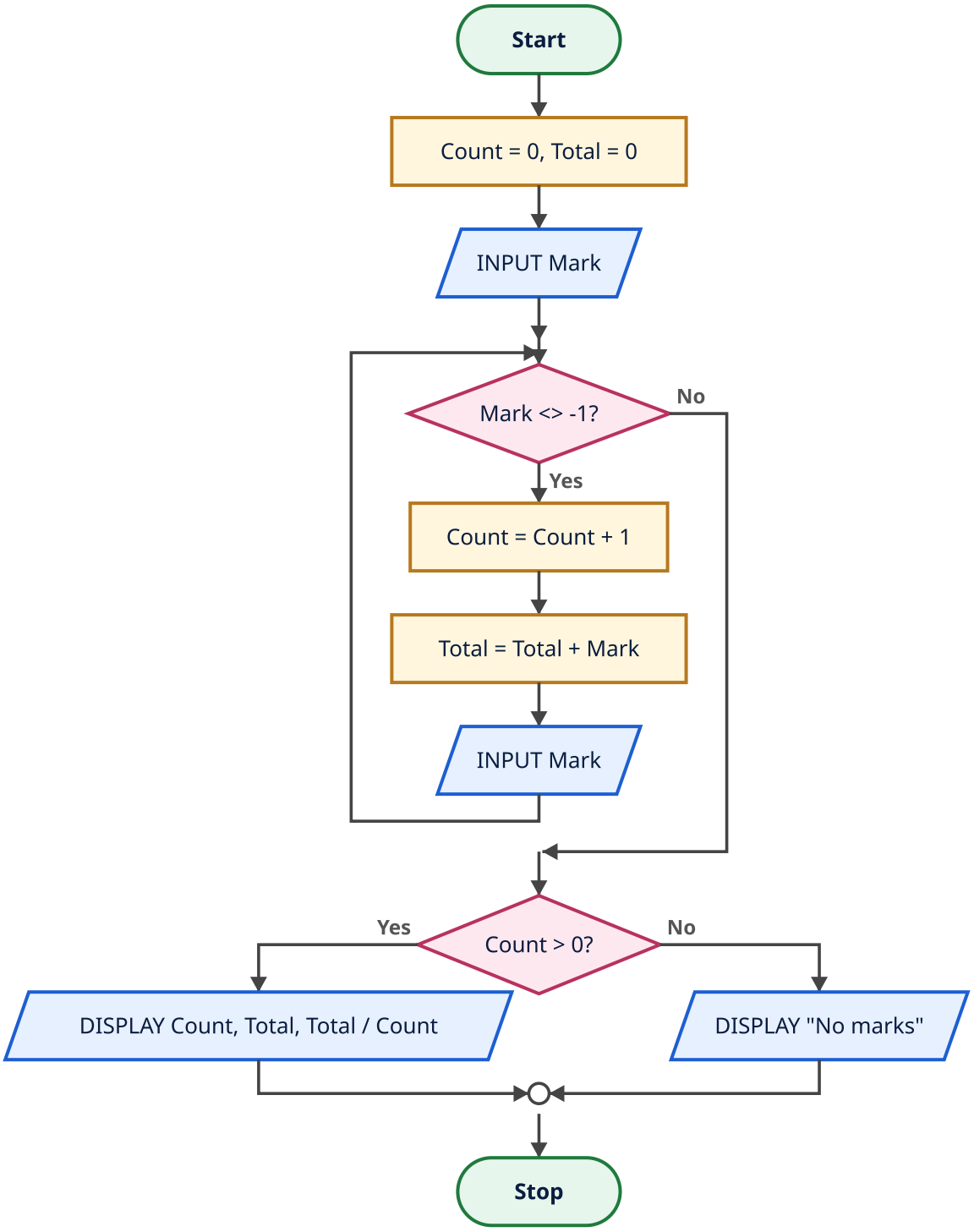


General form of a WHILE loop (test at the beginning)

Pseudocode

```
BEGIN
  Count = 0
  Total = 0
  INPUT Mark
  WHILE Mark <> -1 DO
    Count = Count + 1
    Total = Total + Mark
    INPUT Mark
  ENDWHILE
  IF Count > 0 THEN
    DISPLAY Count, Total, Total / Count
  ELSE
    DISPLAY "No marks entered"
  ENDIF
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – While loop (unknown count): marks until −1

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

While loop (unknown count): marks until −1

Pascal

```
program MarksUntilStop;
var
  mark, count, total: integer;
begin
  count := 0;
  total := 0;
  write('Enter mark (-1 to stop): ');
  readln(mark);
  while mark <> -1 do
  begin
    count := count + 1;
    total := total + mark;
    write('Enter mark (-1 to stop): ');
    readln(mark);
  end;
  if count > 0 then
  begin
    writeln('Count = ', count);
    writeln('Total = ', total);
    writeln('Average = ', total / count:0:2);
  end
  else
    writeln('No marks entered');
end.
```

Python

```
count = 0
total = 0
mark = int(input("Enter mark (-1 to stop): "))
while mark != -1:
    count = count + 1
    total = total + mark
    mark = int(input("Enter mark (-1 to stop): "))
if count > 0:
    print("Count =", count)
    print("Total =", total)
    print(f"Average = {total / count:.2f}")
else:
    print("No marks entered")
```

Sample run 1

```
Enter mark (-1 to stop): 70
Enter mark (-1 to stop): 85
Enter mark (-1 to stop): 64
Enter mark (-1 to stop): -1
Count = 3
Total = 219
Average = 73.00
```

Sample run 2

```
Enter mark (-1 to stop): -1
No marks entered
```

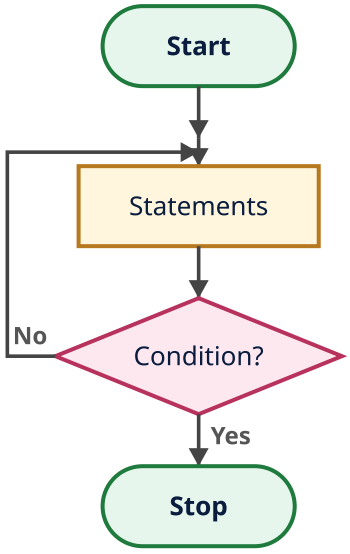
© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Mark	Count	Total
70	1	70
85	2	155
64	3	219
-1	3	219

Count = 3, Total = 219, so Average = 219 ÷ 3 = 73.00. If the first input is −1, the loop body never runs and the program says “No marks entered”.

3. Unknown count, test at the end: REPEAT–UNTIL

A **REPEAT–UNTIL** loop runs the body **first** and checks the condition **at the end**, so the body always runs **at least once**. It stops when the condition becomes **True**. Perfect for checking input: keep asking until a valid mark (0–100) is typed.

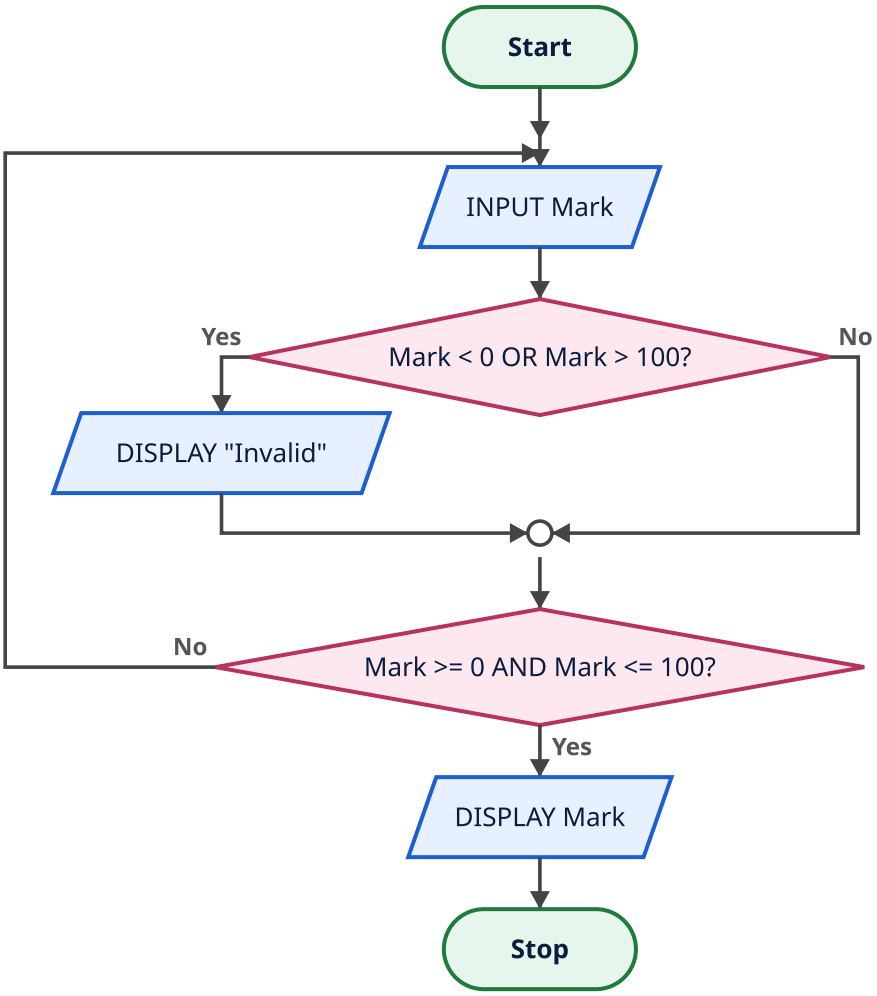


General form of a REPEAT–UNTIL loop (test at the end)

Pseudocode

```
BEGIN
  REPEAT
    INPUT Mark
    IF Mark < 0 OR Mark > 100 THEN
      DISPLAY "Invalid. Try again."
    ENDIF
  UNTIL Mark >= 0 AND Mark <= 100
  DISPLAY Mark
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – Repeat–Until (checked at the end): accept a valid mark

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Repeat–Until (checked at the end): accept a valid mark

Pascal

```
program ValidMark;
var
  mark: integer;
begin
  repeat
    write('Enter mark (0-100): ');
    readln(mark);
    if (mark < 0) or (mark > 100) then
      writeln('Invalid. Try again.');
  until (mark >= 0) and (mark <= 100);
  writeln('Mark accepted: ', mark);
end.
```

Python

```
while True:
    mark = int(input("Enter mark (0-100): "))
    if 0 <= mark <= 100:
        break
    print("Invalid. Try again.")
print("Mark accepted:", mark)
```

Sample run

```
Enter mark (0-100): 120
Invalid. Try again.
Enter mark (0-100): -5
Invalid. Try again.
Enter mark (0-100): 76
Mark accepted: 76
```

💡 Python has **no REPEAT–UNTIL**. We copy it with while True: … and break when the stop condition is reached.

4. Comparing the three loops

	FOR	WHILE	REPEAT–UNTIL
Number of times	Known	Unknown	Unknown
Condition tested	Counter checked at the start	At the beginning	At the end
Minimum runs	0	0	1
Stops when	Counter passes the end value	Condition is False	Condition is True
Pascal	for i := 1 to 10 do	while … do	repeat … until …;
Python	for i in range(1, 11):	while …:	while True: … break

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

5. Converting one loop into another

The same task – display the even numbers 2, 4, 6, 8, 10 – written with each loop. To change a FOR into a WHILE: (1) set the counter **before** the loop, (2) write the condition, (3) **increase the counter inside** the loop. For REPEAT–UNTIL, **reverse** the condition (WHILE i <= 10 becomes UNTIL i > 10).

Pseudocode

```
BEGIN
  FOR i = 1 TO 5
    DISPLAY i * 2
  ENDFOR
END
```

Even numbers 2–10 with FOR

Pascal

```
program EvenFor;
var
  i: integer;
begin
  for i := 1 to 5 do
    writeln(i * 2);
end.
```

Python

```
for i in range(2, 11, 2):
    print(i)
```

Sample run

```
2
4
6
8
10
```

Pseudocode

```
BEGIN
  i = 2
  WHILE i <= 10 DO
    DISPLAY i
    i = i + 2
  ENDWHILE
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Even numbers 2–10 with WHILE

Pascal

```
program EvenWhile;
var
  i: integer;
begin
  i := 2;
  while i <= 10 do
  begin
    writeln(i);
    i := i + 2;
  end;
end.
```

Python

```
i = 2
while i <= 10:
    print(i)
    i = i + 2
```

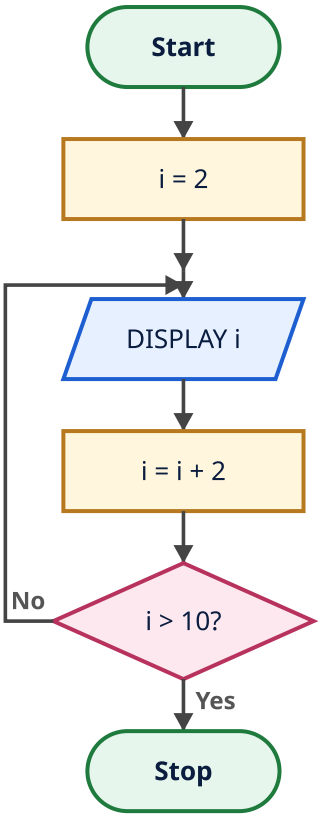
Sample run

```
2
4
6
8
10
```

Pseudocode

```
BEGIN
  i = 2
  REPEAT
    DISPLAY i
    i = i + 2
  UNTIL i > 10
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – Even numbers 2–10 with REPEAT–UNTIL

Even numbers 2–10 with REPEAT–UNTIL

Pascal

```
program EvenRepeat;
var
  i: integer;
begin
  i := 2;
  repeat
    writeln(i);
    i := i + 2;
  until i > 10;
end.
```

Python

```
i = 2
while True:          # Python has no repeat-until
    print(i)
    i = i + 2
    if i > 10:
        break
```

Sample run

```
2
4
6
8
10
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

💡 Danger – an **infinite loop**: if you forget i = i + 2 inside a WHILE loop, i stays 2 for ever and the loop never stops. Always check that something inside the loop moves it towards the end.

Key Words

Loop – Statements that are repeated.	**FOR loop** – A loop that runs a known number of times using a counter.
WHILE loop – A loop that tests its condition at the beginning.	**REPEAT–UNTIL loop** – A loop that tests its condition at the end; runs at least once.
Counter – A variable that counts the loop runs, e.g. i.	**Pre-test loop** – Condition checked before the body (WHILE, FOR).
Post-test loop – Condition checked after the body (REPEAT–UNTIL).	**Sentinel value** – A special value that means “stop”, e.g. −1.
Infinite loop – A loop that never stops.	**Validation** – Checking that input is sensible, e.g. a mark from 0 to 100.
range() – Python function giving numbers for a FOR loop; range(1, 11) is 1 to 10.	**break** – Python word that leaves a loop at once.

Sample Q&A

1. When do we use a FOR loop?

> **Answer:** When we know how many times to repeat.

2. What is the main difference between WHILE and REPEAT–UNTIL?

> **Answer:** WHILE tests at the beginning (may run 0 times); REPEAT–UNTIL tests at the end (runs at least once).

3. What is a sentinel value?

> **Answer:** A special value such as −1 that tells the loop to stop.

4. What is the factorial of 5?

> **Answer:** 120.

5. How many times does for i := 1 to 10 run?

> **Answer:** 10 times.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

6. What causes an infinite loop?

> **Answer:** The condition never changes, e.g. forgetting to increase the counter.

MCQ

1. A loop for a known number of times is...

A. FOR
B. WHILE
C. IF
D. REPEAT–UNTIL

2. Which loop always runs at least once?

A. WHILE
B. FOR
C. None
D. REPEAT–UNTIL

3. Which loop tests the condition at the beginning?

A. CASE
B. WHILE
C. ENDIF
D. REPEAT–UNTIL

4. REPEAT ... UNTIL condition stops when the condition is...

A. zero
B. False
C. True
D. equal to 10

5. In Python, range(1, 11) gives...

A. 1 to 10
B. 0 to 10
C. 1 to 11
D. 0 to 11

6. The output of the factorial program for N = 5 is...

A. Factorial = 60
B. Factorial = 120
C. Factorial = 15
D. Factorial = 25

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

7. Marks 70, 85, 64 then −1 are entered. The average shown is…

A. 54.50
B. 72.67
C. 73.00
D. 219.00

8. In the marks program, −1 is called a…

A. data type
B. counter
C. constant mark
D. sentinel value

9. If the first input is −1, the WHILE loop body runs…

A. 0 times
B. forever
C. 3 times
D. 1 time

10. Python has no REPEAT–UNTIL. We use…

A. while True: … break
B. repeat()
C. do … loop
D. for … until

11. Changing “WHILE i <= 10” to REPEAT–UNTIL gives…

A. UNTIL i < 10
B. UNTIL i <= 10
C. UNTIL i > 10
D. UNTIL i = 2

12. Forgetting to increase the counter inside a WHILE loop causes…

A. a syntax error
B. no output
C. an infinite loop
D. a faster program

13. How many lines does the multiplication-table program print after the input?

A. 9
B. 10
C. 11
D. 7

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

14. Asking again and again until a valid mark is typed is best done with…

A. FOR
B. a constant
C. CASE
D. REPEAT–UNTIL

15. for i := 1 to 5 do write(i * 2) prints…

A. 1 2 3 4 5
B. 2 4 6 8 10
C. 1 3 5 7 9
D. 2 4 6 8

Structured Questions

1. (i) Give one difference between a pre-test and a post-test loop. (2 marks)
(ii) Give one example of each in Pascal. (2 marks) *(4 marks)*

> (i) A pre-test loop checks the condition before the body and may run 0 times; a post-test loop checks after and runs at least once. (ii) while … do (pre-test); repeat … until (post-test).

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

2. Write a program (Pascal or Python) to input a number and display its multiplication table from 1 to 10. (4 marks) *(4 marks)*

Input N (1); FOR i = 1 to 10 (1); calculate N × i (1); display in the form 7 x 1 = 7 (1).

For loop (known count): multiplication table

Pascal

```
program TimesTable;
var
  n, i: integer;
begin
  write('Enter a number: ');
  readln(n);
  for i := 1 to 10 do
    writeln(n, ' x ', i, ' = ', n * i);
end.
```

Python

```
n = int(input("Enter a number: "))
for i in range(1, 11):
    print(n, "x", i, "=", n * i)
```

Sample run

```
Enter a number: 7
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

3. Write pseudocode that keeps reading marks until −1 is entered, then displays the count, total and average. (5 marks) *(5 marks)*

Count = 0, Total = 0 (1); input Mark before the loop (1); WHILE Mark <> −1 add to Count and Total (1); input Mark inside the loop (1); display Count, Total, Total / Count (1).

Pseudocode

```
BEGIN
  Count = 0
  Total = 0
  INPUT Mark
  WHILE Mark <> -1 DO
    Count = Count + 1
    Total = Total + Mark
    INPUT Mark
  ENDWHILE
  IF Count > 0 THEN
    DISPLAY Count, Total, Total / Count
  ELSE
    DISPLAY "No marks entered"
  ENDIF
END
```

Mark	Count	Total
70	1	70
85	2	155
64	3	219
-1	3	219

4. Rewrite this FOR loop as a WHILE loop: FOR i = 1 TO 5 : DISPLAY i * 2 : ENDFOR. (3 marks) *(3 marks)*

i = 1 (1); WHILE i <= 5 DO DISPLAY i * 2 (1); i = i + 1; ENDWHILE (1). (Or the equivalent with i = 2 … i <= 10 … i = i + 2.)

Pseudocode

```
BEGIN
  i = 2
  WHILE i <= 10 DO
    DISPLAY i
    i = i + 2
  ENDWHILE
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

5. Complete a trace table for the factorial program with N = 5 and give the output. (3 marks)
(3 marks)

F = 1, 2, 6, 24, 120 for i = 1 to 5 (2); output Factorial = 120 (1).

i	F = F * i
1	1
2	2
3	6
4	24
5	120

Summary

- FOR: known number of times.
- WHILE: test at the beginning; may run 0 times.
- REPEAT–UNTIL: test at the end; runs at least once.
- REPEAT stops when the condition is True.
- A sentinel (−1) tells the loop to stop.
- Python: for … in range(); while; while True + break.
- Convert: set counter, test, increase inside.
- Always move the loop towards its end – avoid infinite loops.

MCQ answer key – 1.7

1. **A** – Known count → FOR.
2. **D** – Test at the end.
3. **B** – Pre-test.
4. **C** – Until it becomes true.
5. **A** – End not included.
6. **B** – 1×2×3×4×5.
7. **C** – 219 ÷ 3.
8. **D** – Stop signal.
9. **A** – Condition false at start.
10. **A** – while True with break.
11. **C** – Reverse the condition.
12. **C** – Never stops.
13. **B** – i = 1 to 10.
14. **D** – Must ask at least once.
15. **B** – Doubles of 1–5.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

© 2026 schoolict.net (Ruwan Mangala Suraweera). All rights reserved.

G.C.E. O/L Grade 11 ICT · Lesson 1

1.8 Nested Control Structures

Syllabus 10.8. A structure placed **inside** another structure is **nested** – like a small box inside a bigger box. We can put a selection inside a selection, a loop inside a loop, a selection inside a loop, or a loop inside a selection. Real programs use nesting all the time.

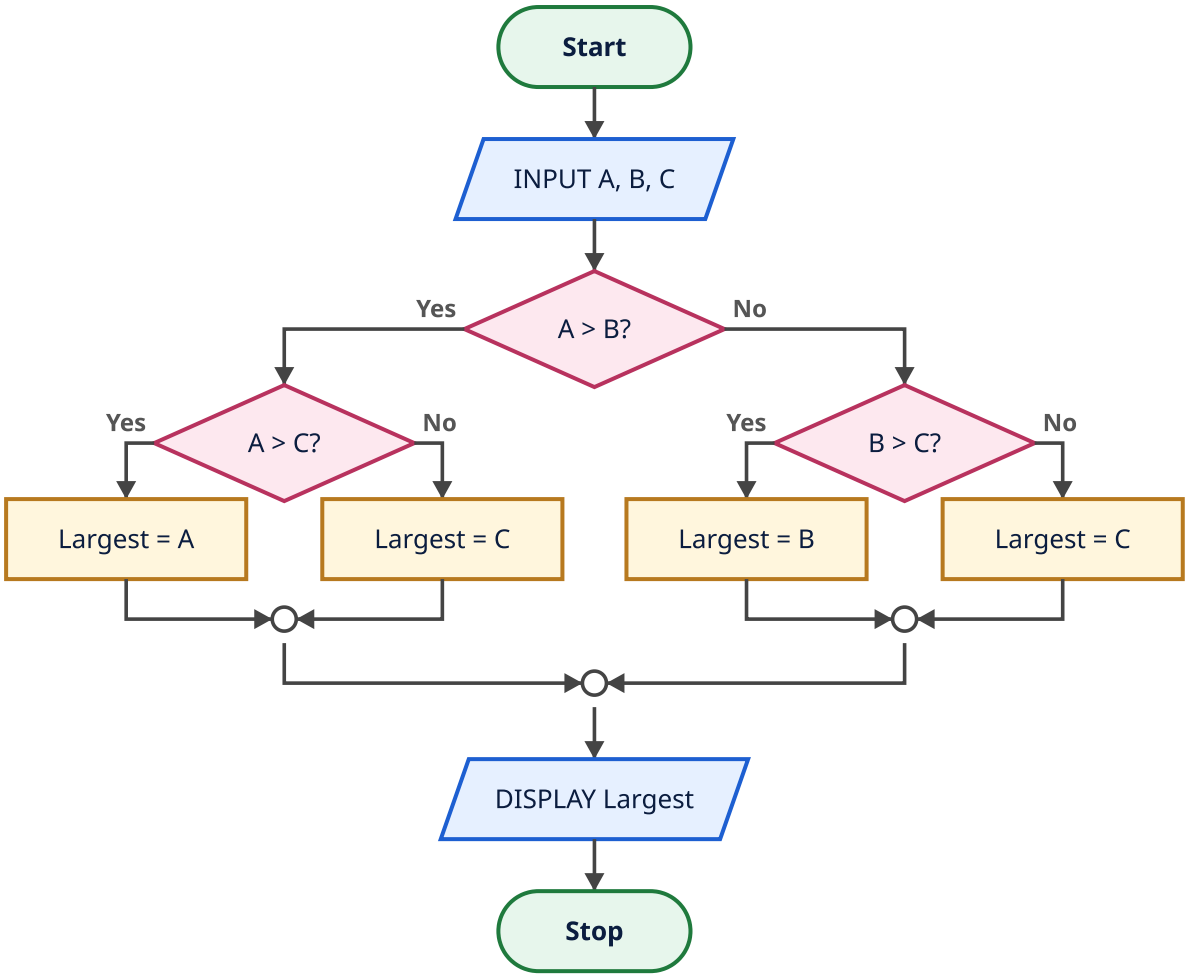
1. Selection inside selection

To find the largest of three numbers A, B and C: first ask “Is A > B?”. Then, **inside each answer**, ask a second question with C. The inner IF runs only when the outer IF reaches it.

Pseudocode

```
BEGIN
  INPUT A, B, C
  IF A > B THEN
    IF A > C THEN
      Largest = A
    ELSE
      Largest = C
    ENDIF
  ELSE
    IF B > C THEN
      Largest = B
    ELSE
      Largest = C
    ENDIF
  ENDIF
  DISPLAY Largest
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – Selection inside selection: largest of three

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Selection inside selection: largest of three

Pascal

```
program Largest3;
var
  a, b, c, largest: integer;
begin
  write('Enter A: ');
  readln(a);
  write('Enter B: ');
  readln(b);
  write('Enter C: ');
  readln(c);
  if a > b then
  begin
    if a > c then
      largest := a
    else
      largest := c;
  end
  else
  begin
    if b > c then
      largest := b
    else
      largest := c;
  end;
  writeln('Largest = ', largest);
end.
```

Python

```
a = int(input("Enter A: "))
b = int(input("Enter B: "))
c = int(input("Enter C: "))
if a > b:
    if a > c:
        largest = a
    else:
        largest = c
else:
    if b > c:
        largest = b
    else:
        largest = c
print("Largest =", largest)
```

Sample run 1

```
Enter A: 12
Enter B: 45
Enter C: 30
Largest = 45
```

Sample run 2

```
Enter A: 50
Enter B: 20
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

```
Enter C: 35
Largest = 50
```

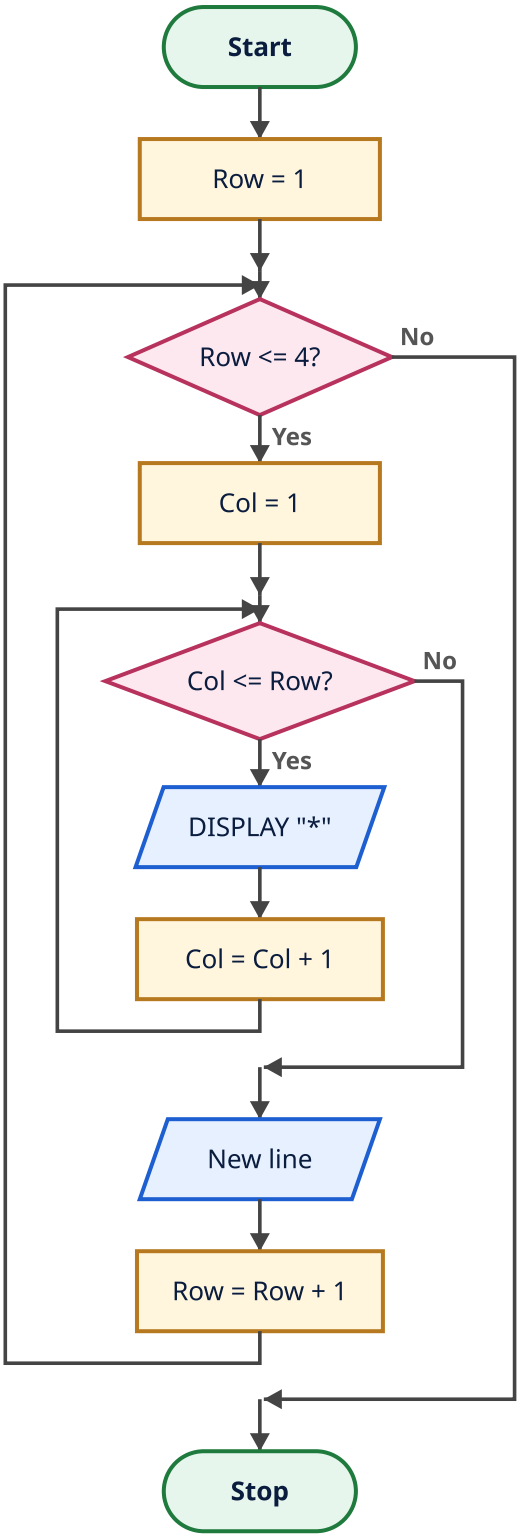
2. Iteration inside iteration

An **inner loop** runs **completely** for **every single run** of the outer loop. In the star triangle, the outer loop chooses the row; the inner loop prints that many stars.

Pseudocode

```
BEGIN
  FOR Row = 1 TO 4
    FOR Col = 1 TO Row
      DISPLAY "*" (same line)
    ENDFOR
    Move to a new line
  ENDFOR
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – Iteration inside iteration: star triangle

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Iteration inside iteration: star triangle

Pascal

```
program Stars;
var
  row, col: integer;
begin
  for row := 1 to 4 do
  begin
    for col := 1 to row do
      write('*');
    writeln;
  end;
end.
```

Python

```
for row in range(1, 5):
    for col in range(1, row + 1):
        print("*", end="")
    print()
```

Sample run

```
*
**
***
****
```

Row	Inner loop runs (times)	Printed
1	1	*
2	2	**
3	3	***
4	4	****

💡 Total stars = 1 + 2 + 3 + 4 = 10. If the outer loop runs 4 times and the inner loop always ran 3 times, the inner body would run 4 × 3 = 12 times.

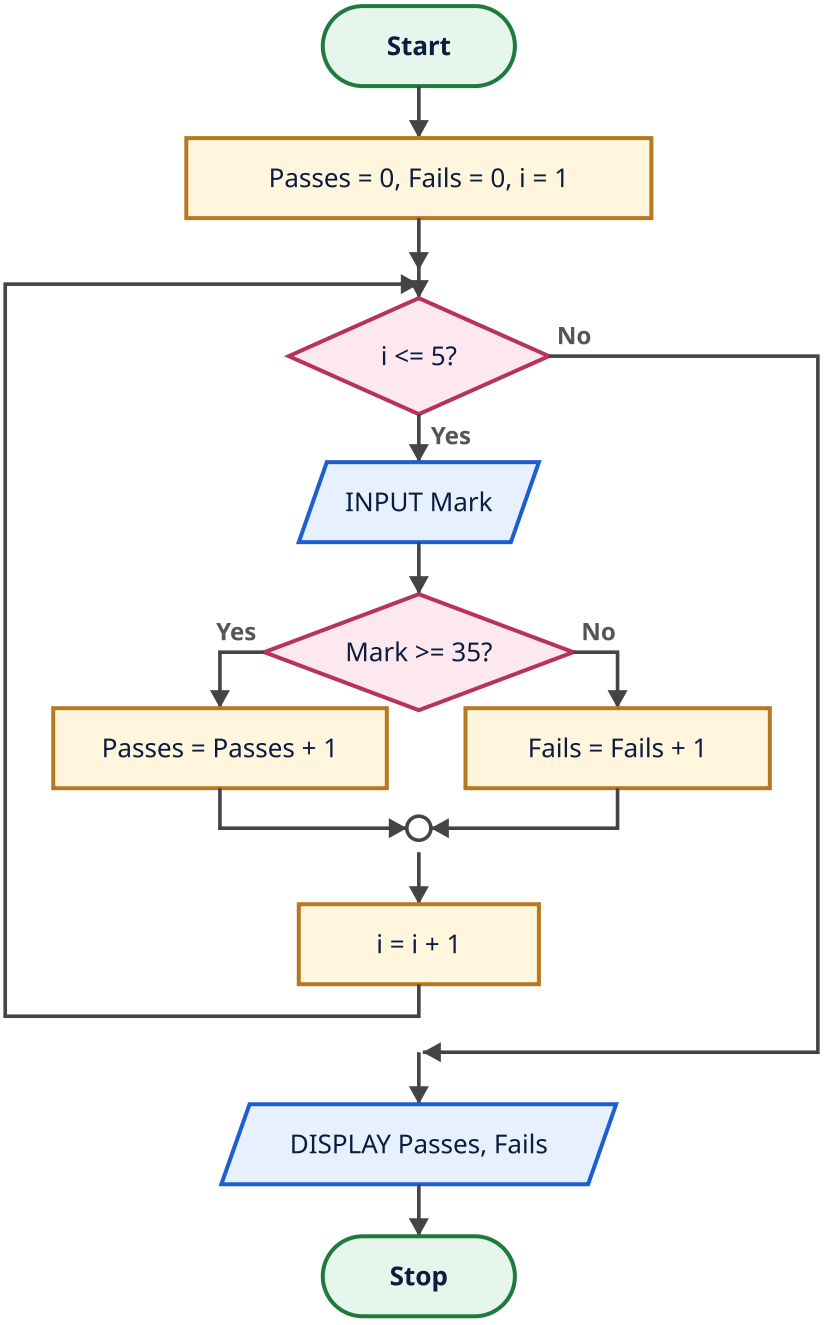
3. Selection inside iteration

A loop reads 5 marks. **Inside** the loop, an IF decides whether each mark is a pass (≥ 35) or a fail, and counts them.

Pseudocode

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

```
BEGIN
  Passes = 0
  Fails = 0
  FOR i = 1 TO 5
    INPUT Mark
    IF Mark >= 35 THEN
      Passes = Passes + 1
    ELSE
      Fails = Fails + 1
    ENDIF
  ENDFOR
  DISPLAY Passes, Fails
END
```



Flowchart – Selection inside iteration: count passes and fails

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Selection inside iteration: count passes and fails

Pascal

```
program PassCount;
var
  i, mark, passes, fails: integer;
begin
  passes := 0;
  fails := 0;
  for i := 1 to 5 do
  begin
    write('Enter mark ', i, ': ');
    readln(mark);
    if mark >= 35 then
      passes := passes + 1
    else
      fails := fails + 1;
  end;
  writeln('Pass = ', passes);
  writeln('Fail = ', fails);
end.
```

Python

```
passes = 0
fails = 0
for i in range(1, 6):
    mark = int(input(f"Enter mark {i}: "))
    if mark >= 35:
        passes = passes + 1
    else:
        fails = fails + 1
print("Pass =", passes)
print("Fail =", fails)
```

Sample run

```
Enter mark 1: 45
Enter mark 2: 30
Enter mark 3: 78
Enter mark 4: 22
Enter mark 5: 60
Pass = 3
Fail = 2
```

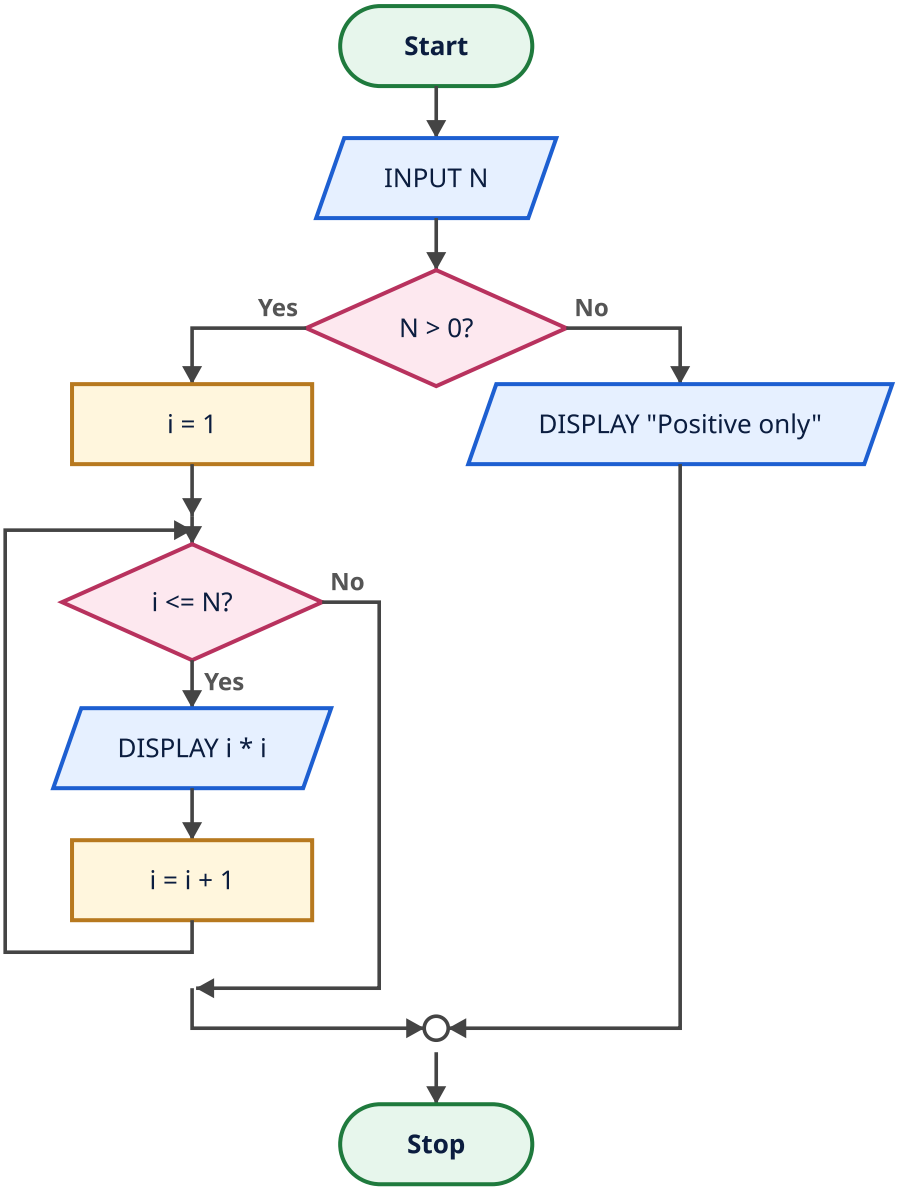
4. Iteration inside selection

First check that the input is positive. **Only if it is**, run a loop that displays the squares 1^2, 2^2, ...

Pseudocode

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

```
BEGIN
  INPUT N
  IF N > 0 THEN
    FOR i = 1 TO N
      DISPLAY i * i
    ENDFOR
  ELSE
    DISPLAY "Please enter a positive number"
  ENDIF
END
```



Flowchart – Iteration inside selection: squares only for a positive number

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Iteration inside selection: squares only for a positive number

Pascal

```
program Squares;
var
  n, i: integer;
begin
  write('How many squares? ');
  readln(n);
  if n > 0 then
  begin
    for i := 1 to n do
      writeln('Square of ', i, ' = ', i * i);
  end
  else
    writeln('Please enter a positive number');
end.
```

Python

```
n = int(input("How many squares? "))
if n > 0:
    for i in range(1, n + 1):
        print("Square of", i, "=", i * i)
else:
    print("Please enter a positive number")
```

Sample run 1

```
How many squares? 4
Square of 1 = 1
Square of 2 = 4
Square of 3 = 9
Square of 4 = 16
```

Sample run 2

```
How many squares? -2
Please enter a positive number
```

5. Tips for nested code

- **Indent** each inner level a little more, so you can see which statement belongs where.
- Close the **inner** structure before the outer one (ENDIF before ENDFOR).
- In Pascal, use **begin … end** when a part has more than one statement.
- Use different counter names for nested loops (Row and Col, i and j).

Key Words

Nesting – Placing one control structure inside another.

Nested IF – An IF inside another IF.

Nested loop – A loop inside another loop.

Outer loop – The loop that contains another loop.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Inner loop – The loop inside; it runs fully for each outer run.

Indentation – Spaces that show which level a statement belongs to.

begin … end – Pascal words that group several statements into one block.

Largest – The biggest value in a group.

Pass count – A counter that adds 1 for every pass.

Square – A number multiplied by itself, e.g. 4 × 4 = 16.

Positive number – A number greater than 0.

Level – How deep a structure is inside others.

Sample Q&A

1. What is a nested structure?

> **Answer:** A control structure placed inside another one.

2. For A = 12, B = 45, C = 30, what does the largest-of-three program display?

> **Answer:** Largest = 45.

3. If the outer loop runs 3 times and the inner loop runs 4 times each time, how many times does the inner body run?

> **Answer:** 12 times.

4. Marks 45, 30, 78, 22, 60 – how many passes and fails?

> **Answer:** Pass = 3, Fail = 2.

5. What does the squares program show for −2?

> **Answer:** Please enter a positive number.

6. Why do we indent nested code?

> **Answer:** To see clearly which statements belong to which structure.

MCQ

1. A loop placed inside another loop is a…

A. constant
B. sentinel
C. nested loop
D. case

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

2. For A = 50, B = 20, C = 35, the largest is…

A. 50
B. 35
C. 105
D. 20

3. In the largest-of-three flowchart, the first question is…

A. A > C?
B. C > 0?
C. B > C?
D. A > B?

4. Outer loop runs 4 times, inner loop 3 times each. Inner body runs…

A. 3 times
B. 12 times
C. 4 times
D. 7 times

5. The last line of the star triangle (Row = 4) is…

A. *
B. ****
C. **
D. ***

6. Total stars in the 4-row triangle…

A. 10
B. 4
C. 16
D. 8

7. In the star program, the inner loop runs while…

A. Col = 0
B. Row > Col + 5
C. Col <= Row
D. Row <= 4 only

8. Marks 45, 30, 78, 22, 60 give Pass = …

A. 2
B. 4
C. 5
D. 3

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

9. Counting passes inside a loop is an example of…

A. selection inside iteration
B. iteration inside selection
C. a nested array
D. sequence only

10. The squares program checks the number first, then loops. This is…

A. selection inside iteration
B. a case statement
C. a loop inside a loop
D. iteration inside selection

11. For input 4, the last line of the squares program is…

A. Square of 4 = 8
B. Square of 4 = 16
C. Square of 3 = 9
D. Square of 16 = 4

12. Which should be closed first?

A. the outer structure
B. the program
C. the inner structure
D. any one

13. In Pascal, several statements inside an IF must be grouped with…

A. [ ]
B. begin … end
C. { }
D. ( )

14. Good names for two nested loop counters are…

A. both Total
B. x and x
C. Row and Col
D. both i

15. Nesting means…

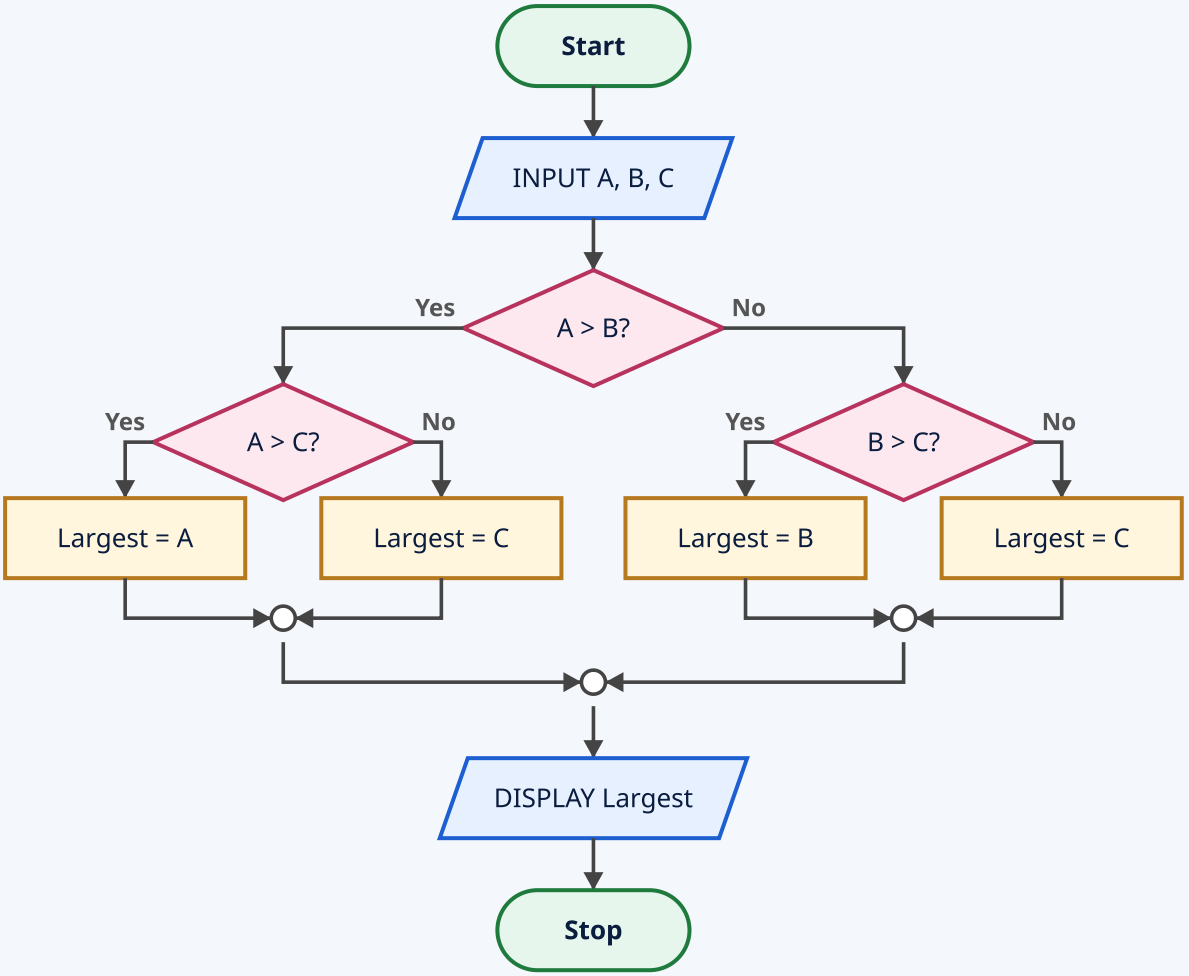
A. putting one structure inside another
B. using only sequence
C. deleting a loop
D. joining two programs

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Structured Questions

1. Draw a flowchart to input three numbers A, B and C and display the largest, using nested IFs. (5 marks) *(5 marks)*

Start/input (1); A > B? (1); Yes → A > C? (1); No → B > C? (1); assign Largest and display, Stop (1).



Flowchart – Selection inside selection: largest of three

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

2. Write a program (Pascal or Python) that uses nested loops to print the triangle *, , *, **. (4 marks)** *(4 marks)*

Outer loop Row 1 to 4 (1); inner loop Col 1 to Row (1); print * without new line (1); new line after the inner loop (1).

Iteration inside iteration: star triangle

Pascal

```
program Stars;
var
  row, col: integer;
begin
  for row := 1 to 4 do
  begin
    for col := 1 to row do
      write('*');
    writeln;
  end;
end.
```

Python

```
for row in range(1, 5):
    for col in range(1, row + 1):
        print("*", end="")
    print()
```

Sample run

```
*
**
***
****
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

3. Write pseudocode to read 5 marks and count how many are passes (≥ 35) and how many are fails. (5 marks) *(5 marks)*

PassCount = 0, FailCount = 0 (1); FOR k = 1 TO 5 (1); INPUT Mark (1); IF Mark >= 35 add 1 to PassCount ELSE to FailCount (1); display both (1).

Pseudocode

```
BEGIN
  Passes = 0
  Fails = 0
  FOR i = 1 TO 5
    INPUT Mark
    IF Mark >= 35 THEN
      Passes = Passes + 1
    ELSE
      Fails = Fails + 1
    ENDIF
  ENDFOR
  DISPLAY Passes, Fails
END
```

4. Name the four kinds of nesting and give an example of each. (4 marks) *(4 marks)*

Selection in selection – largest of three; iteration in iteration – star triangle; selection in iteration – count passes; iteration in selection – squares only for a positive number (1 each).

5. Complete a trace table for the star program (4 rows) showing Row, how many times the inner loop runs, and what is printed. (3 marks) *(3 marks)*

Row 1 → 1 → *; Row 2 → 2 → **; **Row 3** → **3** → ***; Row 4 → 4 → **** (3 for all correct).

Row	Inner loop runs (times)	Printed
1	1	*
2	2	**
3	3	***
4	4	****

Summary

- Nesting = a structure inside another.
- Nested IF: largest of three.
- Nested loop: inner loop runs fully each time.
- Inner runs = outer × inner (fixed sizes).
- IF inside a loop: count passes.
- Loop inside an IF: run only when valid.
- Indent every level; close inner first.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

- Pascal: begin … end groups statements.

MCQ answer key – 1.8

1. **C** – Nested.
2. **A** – A is biggest.
3. **D** – Outer IF.
4. **B** – 4 × 3.
5. **B** – 4 stars.
6. **A** – 1+2+3+4.
7. **C** – Inner condition.
8. **D** – 45, 78, 60.
9. **A** – IF inside FOR.
10. **D** – FOR inside IF.
11. **B** – 4 × 4.
12. **C** – Inner first.
13. **B** – begin … end.
14. **C** – Different names.
15. **A** – Definition.

© 2026 schoolict.net (Ruwan Mangala Suraweera). All rights reserved.

G.C.E. O/L Grade 11 ICT · Lesson 1

1.9 One-Dimensional Arrays

Syllabus 10.9. Imagine storing the marks of 40 students in 40 separate variables – Mark1, Mark2 ... Mark40. Very messy! An **array** stores many values of the **same data type** under **one name**. Each value is an **element**, and we reach it with its **index** (position number), like Marks[3]. Arrays are kept in **contiguous** (side-by-side) memory, so any element can be reached **directly** (random access).

1. Why do we need arrays?

- **One name** for many values – easy to read.
- Use a **loop** to process all elements with a few lines.
- Easy to find the total, average, highest and to search or sort.
- To change 5 students to 40, change one number only.

2. Index, elements and memory

Think of a row of numbered lockers in a school corridor. All lockers are **next to each other** (contiguous), and you can open locker 4 **directly** without opening 1, 2 and 3 first (random access). The array Marks with 5 elements:

Pascal index	1	2	3	4	5
Python index	0	1	2	3	4
Value	65	80	45	90	70

💡 **Index starts:** in Pascal we choose it (array[1..5] starts at 1). In **Python** a list always starts at **0**, so the first mark is marks[0] and the fifth is marks[4]. Pseudocode in O/L normally starts at 1.

3. Declare, assign and access

Action	Pascal	Python
Declare (5 integers)	var Marks: array[1..5] of integer;	marks = [0, 0, 0, 0, 0]
Assign a value	Marks[3] := 45;	marks[2] = 45
Access (read) a value	writeln(Marks[3]);	print(marks[2])
Fill with a loop	for i := 1 to 5 do readln(Marks[i]);	for i in range(5): marks[i] = int(input())

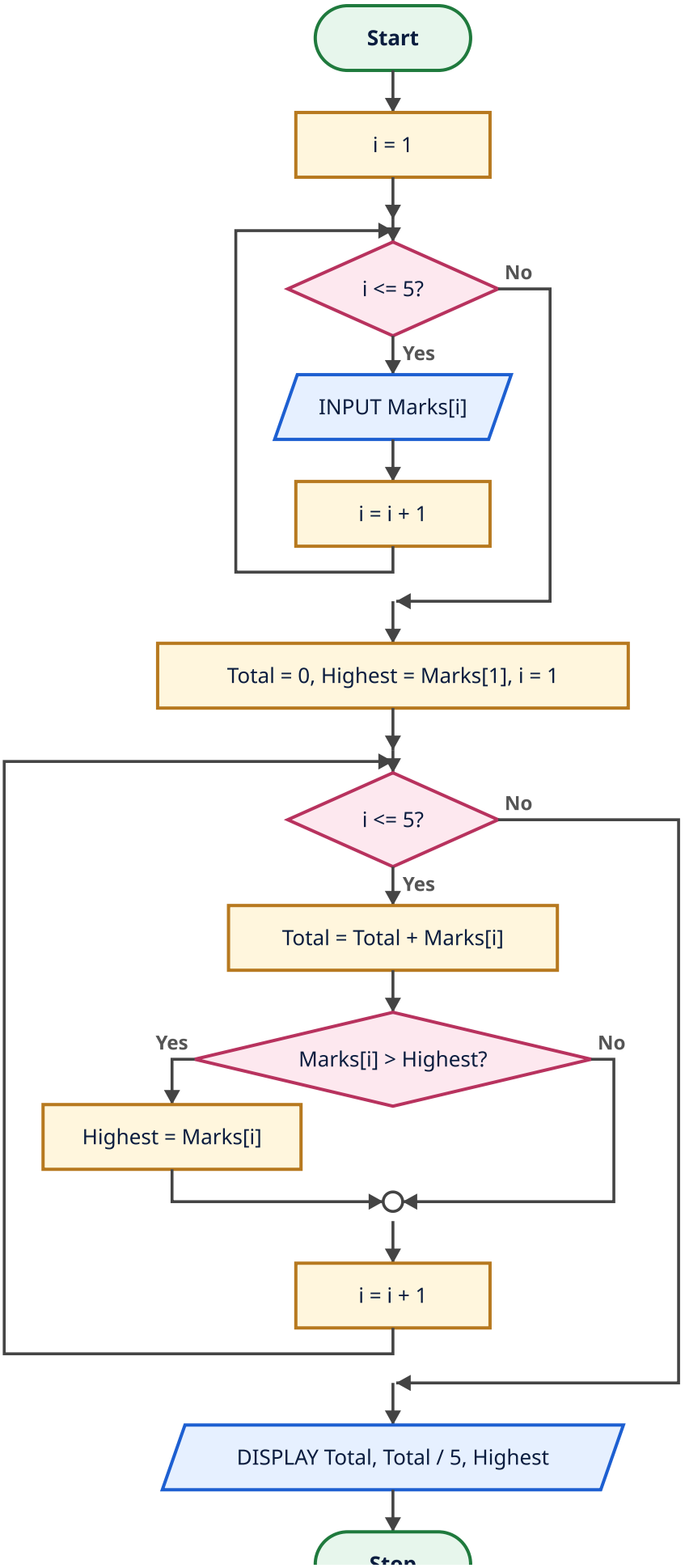
4. Example: total, average and highest

Pseudocode

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

```
BEGIN
  FOR i = 1 TO 5
    INPUT Marks[i]
  ENDFOR
  Total = 0
  Highest = Marks[1]
  FOR i = 1 TO 5
    Total = Total + Marks[i]
    IF Marks[i] > Highest THEN
      Highest = Marks[i]
    ENDIF
  ENDFOR
  DISPLAY Total, Total / 5, Highest
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Stop

Flowchart – Array: store 5 marks, then total, average and highest

Array: store 5 marks, then total, average and highest

Pascal

```
program MarksArray;
var
  marks: array[1..5] of integer;
  i, total, highest: integer;
begin
  for i := 1 to 5 do
  begin
    write('Enter mark ', i, ': ');
    readln(marks[i]);
  end;
  total := 0;
  highest := marks[1];
  for i := 1 to 5 do
  begin
    total := total + marks[i];
    if marks[i] > highest then
      highest := marks[i];
  end;
  writeln('Total = ', total);
  writeln('Average = ', total / 5:0:2);
  writeln('Highest = ', highest);
end.
```

Python

```
marks = [0, 0, 0, 0, 0]          # 5 elements, index 0 to 4
for i in range(5):
    marks[i] = int(input(f"Enter mark {i + 1}: "))
total = 0
highest = marks[0]
for i in range(5):
    total = total + marks[i]
    if marks[i] > highest:
        highest = marks[i]
print("Total =", total)
print(f"Average = {total / 5:.2f}")
print("Highest =", highest)
```

Sample run

```
Enter mark 1: 65
Enter mark 2: 80
Enter mark 3: 45
Enter mark 4: 90
Enter mark 5: 70
Total = 350
Average = 70.00
Highest = 90
```

i	Marks[i]	Total	Marks[i] > Highest ?	Highest

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

1	65	65	No	65
2	80	145	Yes	80
3	45	190	No	80
4	90	280	Yes	90
5	70	350	No	90

Total = 350 and Highest = 90, the same as the real program output. Average = 350 ÷ 5 = 70.00.

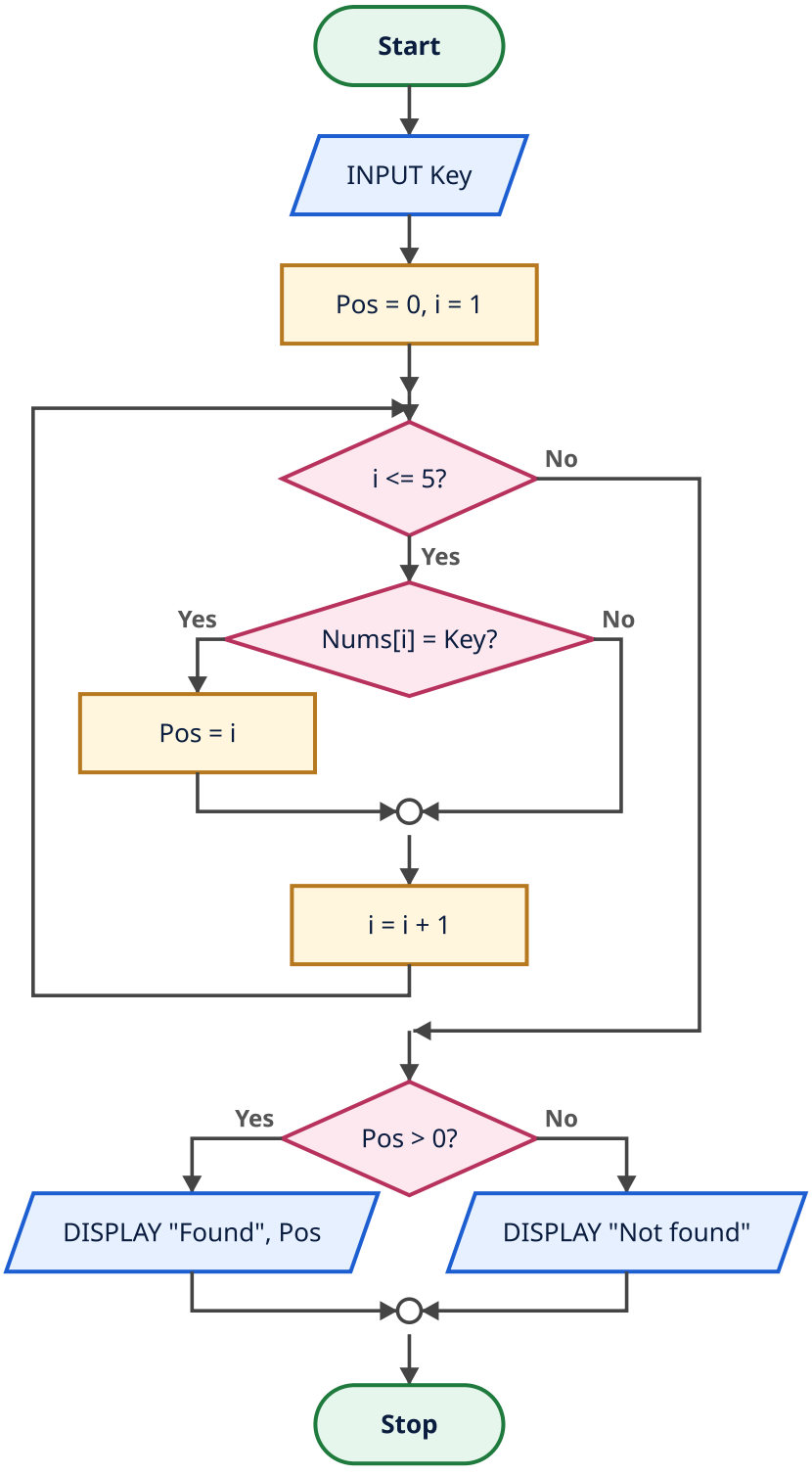
5. Example: searching an array (linear search)

The array Nums holds 12, 45, 7, 30, 18. To find a number, check each element **from the first to the last**. This is called a **linear search**.

Pseudocode

```
BEGIN
  Nums = [12, 45, 7, 30, 18]
  INPUT Key
  Pos = 0
  FOR i = 1 TO 5
    IF Nums[i] = Key THEN
      Pos = i
    ENDIF
  ENDFOR
  IF Pos > 0 THEN
    DISPLAY "Found at position ", Pos
  ELSE
    DISPLAY "Not found"
  ENDIF
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved



Flowchart – Array: search for a number

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Array: search for a number

Pascal

```
program SearchArray;
var
  nums: array[1..5] of integer;
  i, key, pos: integer;
begin
  nums[1] := 12;
  nums[2] := 45;
  nums[3] := 7;
  nums[4] := 30;
  nums[5] := 18;
  write('Enter number to find: ');
  readln(key);
  pos := 0;
  for i := 1 to 5 do
    if nums[i] = key then
      pos := i;
  if pos > 0 then
    writeln('Found at position ', pos)
  else
    writeln('Not found');
end.
```

Python

```
nums = [12, 45, 7, 30, 18]
key = int(input("Enter number to find: "))
pos = 0
for i in range(5):
    if nums[i] == key:
        pos = i + 1      # position counted from 1
if pos > 0:
    print("Found at position", pos)
else:
    print("Not found")
```

Sample run 1

```
Enter number to find: 30
Found at position 4
```

Sample run 2

```
Enter number to find: 50
Not found
```

i	Nums[i]	Nums[i] = 30 ?	Pos
1	12	No	0
2	45	No	0
3	7	No	0
4	30	Yes	4
5	18	No	4

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

6. Finding a position

Temperatures (°C) in Colombo for a week are stored in an array. The program keeps the **index** of the highest value, not just the value, so it can tell us **which day** was hottest.

Position of the highest temperature

Pascal

```
program HottestDay;
var
  t: array[1..7] of integer;
  i, best: integer;
begin
  t[1] := 31; t[2] := 33; t[3] := 30; t[4] := 34;
  t[5] := 32; t[6] := 29; t[7] := 33;
  best := 1;
  for i := 2 to 7 do
    if t[i] > t[best] then
      best := i;
  writeln('Day ', best, ': ', t[best]);
end.
```

Python

```
t = [31, 33, 30, 34, 32, 29, 33]
best = 0
for i in range(1, 7):
    if t[i] > t[best]:
        best = i
print("Day " + str(best + 1) + ":", t[best])
```

Sample run

```
Day 4: 34
```

Key Words

Array – A collection of values of the same type under one name.

One-dimensional array – An array that is a single row (list) of elements.

Element – One value stored in an array.

Index – The position number of an element, e.g. 3 in Marks[3].

Contiguous memory – Elements stored side by side in memory.

Random (direct) access – Reaching any element at once using its index.

Declare – Creating the array with its name, size and type.

Assign – Storing a value in an element, Marks[3] := 45.

Access – Reading the value of an element.

List (Python) – Python's array-like type; index starts at 0.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Linear search – Checking elements one by one from the start.

Array size – The number of elements in an array.

Sample Q&A

1. What is an array?

> **Answer:** A group of values of the same data type stored under one name.

2. What is an index?

> **Answer:** The position number used to reach an element.

3. What does “random access” mean for arrays?

> **Answer:** Any element can be reached directly with its index.

4. Declare an integer array of 5 elements in Pascal.

> **Answer:** var Marks: array[1..5] of integer;

5. In Python, what is the index of the first element?

> **Answer:** 0.

6. For marks 65, 80, 45, 90, 70 what is the highest?

> **Answer:** 90.

MCQ

1. An array stores many values of...

A. the same data type
B. only text
C. only Boolean
D. different data types always

2. One value inside an array is called an...

A. operator
B. element
C. index
D. identifier

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

3. The position number of an element is its…

A. index
B. value
C. type
D. size

4. Array elements are stored in…

A. separate files
B. random places on the disk
C. contiguous memory
D. the printer

5. Reaching Marks[4] without reading Marks[1..3] is called…

A. nesting
B. sequential access
C. random (direct) access
D. compiling

6. Correct Pascal declaration of an array of 5 integers:

A. A = [5]
B. array A of 5;
C. var A: integer[5];
D. var A: array[1..5] of integer;

7. In Python, the first element of marks is…

A. marks(1)
B. marks[0]
C. marks[1]
D. marks{0}

8. A Python list of 5 elements has its last index…

A. 4
B. 1
C. 6
D. 5

9. Marks = 65, 80, 45, 90, 70 (Pascal). Marks[3] is…

A. 90
B. 45
C. 3
D. 80

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

10. The total of 65, 80, 45, 90 and 70 is...

A. 90
B. 340
C. 350
D. 70

11. Nums = 12, 45, 7, 30, 18. Searching for 30 gives...

A. Found at position 4
B. Not found
C. Found at position 3
D. Found at position 30

12. Searching for 50 in the same array gives...

A. Found at position 1
B. An error
C. Not found
D. Found at position 5

13. Checking elements one by one from the start is a...

A. random search
B. nested IF
C. constant
D. linear search

14. Which statement assigns 45 to the third element in Python?

A. 45 = marks[2]
B. marks = 45
C. marks[3] = 45
D. marks[2] = 45

15. Temperatures 31, 33, 30, 34, 32, 29, 33. The hottest day is...

A. Day 6: 29
B. Day 4: 34
C. Day 7: 33
D. Day 2: 33

Structured Questions

1. (i) What is a one-dimensional array? (1 mark)
(ii) Give two advantages of using an array instead of separate variables. (2 marks) *(3 marks)*

> (i) A single row of elements of the same type under one name. (ii) One name for many values; can process all with a loop; easy to change size (any two).

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

2. Explain the terms (a) index (b) contiguous memory (c) random access. (3 marks) *(3 marks)*

> (a) Position number of an element. (b) Elements are stored side by side in memory. (c) Any element can be reached directly using its index.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

3. Write a program (Pascal or Python) to read 5 marks into an array and display the total, average and highest mark. (6 marks) *(6 marks)*

Declare array (1); loop to read (1); total = 0, highest = first element (1); loop adding and comparing (2); display total, average, highest (1).

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Array: store 5 marks, then total, average and highest

Pascal

```
program MarksArray;
var
  marks: array[1..5] of integer;
  i, total, highest: integer;
begin
  for i := 1 to 5 do
  begin
    write('Enter mark ', i, ': ');
    readln(marks[i]);
  end;
  total := 0;
  highest := marks[1];
  for i := 1 to 5 do
  begin
    total := total + marks[i];
    if marks[i] > highest then
      highest := marks[i];
  end;
  writeln('Total = ', total);
  writeln('Average = ', total / 5:0:2);
  writeln('Highest = ', highest);
end.
```

Python

```
marks = [0, 0, 0, 0, 0]          # 5 elements, index 0 to 4
for i in range(5):
    marks[i] = int(input(f"Enter mark {i + 1}: "))
total = 0
highest = marks[0]
for i in range(5):
    total = total + marks[i]
    if marks[i] > highest:
        highest = marks[i]
print("Total =", total)
print(f"Average = {total / 5:.2f}")
print("Highest =", highest)
```

Sample run

```
Enter mark 1: 65
Enter mark 2: 80
Enter mark 3: 45
Enter mark 4: 90
Enter mark 5: 70
Total = 350
Average = 70.00
Highest = 90
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

4. Nums = 12, 45, 7, 30, 18. Complete a trace table for a linear search for 30 and give the output. (4 marks) *(4 marks)*

> i = 1…5 with Nums[i] = 30? No, No, No, Yes, No; Pos = 0, 0, 0, 4, 4 (3); output Found at position 4 (1).

i	Nums[i]	Nums[i] = 30 ?	Pos
1	12	No	0
2	45	No	0
3	7	No	0
4	30	Yes	4
5	18	No	4

5. Write Pascal and Python statements to (a) store 45 in the third element of Marks (b) display the first element. (4 marks) *(4 marks)*

> (a) Pascal Marks[3] := 45; Python marks[2] = 45. (b) Pascal writeln(Marks[1]); Python print(marks[0]) (1 each).

Summary

- Array: many values, same type, one name.
- Element = one value; index = its position.
- Contiguous memory → random (direct) access.
- Pascal: array[1..5] of integer.
- Python list index starts at 0.
- Use a loop to fill and process an array.
- Total, average, highest with one loop.
- Linear search checks elements one by one.

MCQ answer key – 1.9

1. **A** – Same type.
2. **B** – Element.
3. **A** – Index.
4. **C** – Side by side.
5. **C** – Direct.
6. **D** – Pascal syntax.
7. **B** – Index from 0.
8. **A** – 0 to 4.
9. **B** – Third element.
10. **C** – Sum.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

11. **A** – Fourth element.
12. **C** – Not in the array.
13. **D** – Linear.
14. **D** – Index 2 = third.
15. **B** – 34 is 4th.

© 2026 schoolict.net (Ruwan Mangala Suraweera). All rights reserved.

G.C.E. O/L Grade 11 ICT · Lesson 1

1.10 Sub-programs: Functions and Procedures

Syllabus 10.10. A **sub-program** is a small, named part of a program that does **one job**. We write it once and **call** (use) it whenever we need it – like a recipe for pol sambol that you can follow every day without rewriting it. Big programs are built from many small sub-programs.

1. Why use sub-programs?

Reusability
Write once, call many times. No need to copy the same code again.

Readability
Names like PrintLine and GetGrade make the main program easy to understand.

Maintainability
To fix or change a job, change it in one place only.

Teamwork and testing
Different people can write and test different sub-programs.

2. Two kinds of sub-programs

	Value-returning	Non-value-returning
Pascal name	function	procedure
Python	def … with return value	def … without return value
Gives back	one value (e.g. a grade)	nothing – it just does a job
Example	GetGrade(82) → 'A'	PrintLine

3. Parameters and arguments

Values we send into a sub-program are **parameters**. In area(length, width), length and width are parameters. When we call area(5, 4), the values 5 and 4 are the **arguments**.

4. A value-returning function

GetGrade receives a mark and **returns** a grade letter, using the O/L grade limits (A 75, B 65, C 50, S 35).

Pseudocode

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

```
FUNCTION GetGrade(Marks)
  IF Marks >= 75 THEN RETURN "A"
  ELSE IF Marks >= 65 THEN RETURN "B"
  ELSE IF Marks >= 50 THEN RETURN "C"
  ELSE IF Marks >= 35 THEN RETURN "S"
  ELSE RETURN "W"
  ENDIF
ENDFUNCTION

BEGIN
  FOR k = 1 TO 3
    INPUT M
    DISPLAY GetGrade(M)
  ENDFOR
END
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Value-returning sub-program (function): GetGrade

Pascal

```
program GradeFunction;
var
  k, m: integer;

function GetGrade(marks: integer): char;
begin
  if marks >= 75 then
    GetGrade := 'A'
  else if marks >= 65 then
    GetGrade := 'B'
  else if marks >= 50 then
    GetGrade := 'C'
  else if marks >= 35 then
    GetGrade := 'S'
  else
    GetGrade := 'W';
end;

begin
  for k := 1 to 3 do
  begin
    write('Enter mark: ');
    readln(m);
    writeln('Grade: ', GetGrade(m));
  end;
end.
```

Python

```
def get_grade(marks):
    if marks >= 75:
        return "A"
    elif marks >= 65:
        return "B"
    elif marks >= 50:
        return "C"
    elif marks >= 35:
        return "S"
    else:
        return "W"

for k in range(3):
    m = int(input("Enter mark: "))
    print("Grade:", get_grade(m))
```

Sample run

```
Enter mark: 82
Grade: A
Enter mark: 47
Grade: S
Enter mark: 66
Grade: B
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

> 💡 In Pascal, the result is given by assigning to the function name: GetGrade := 'A'. In Python we write return "A".

5. Procedures (non-value-returning)

A shop bill uses PrintLine three times and ShowItem for each item. PrintLine has **no parameters**; ShowItem has three.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Non-value-returning sub-programs (procedures): a shop bill

Pascal

```
program BillProcedures;

procedure PrintLine;
begin
  writeln('------------------------');
end;

procedure ShowItem(item: string; price, qty: integer);
begin
  writeln(item, ': Rs. ', price * qty);
end;

begin
  PrintLine;
  writeln('SAMPLE SHOP BILL');
  PrintLine;
  ShowItem('Bread', 120, 2);
  ShowItem('Milk packet', 280, 1);
  ShowItem('Eggs', 55, 6);
  PrintLine;
end.
```

Python

```
def print_line():
    print("-" * 24)

def show_item(item, price, qty):
    print(item + ": Rs.", price * qty)

print_line()
print("SAMPLE SHOP BILL")
print_line()
show_item("Bread", 120, 2)
show_item("Milk packet", 280, 1)
show_item("Eggs", 55, 6)
print_line()
```

Sample run

```
------------------------
SAMPLE SHOP BILL
------------------------
Bread: Rs. 240
Milk packet: Rs. 280
Eggs: Rs. 330
------------------------
```

6. Calling a function many times

A function can be used inside an expression, just like a number.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

A function with two parameters: area of a rectangle

Pascal

```
program RectArea;

function Area(length, width: integer): integer;
begin
  Area := length * width;
end;

begin
  writeln('Room 1 = ', Area(5, 4));
  writeln('Room 2 = ', Area(6, 3));
  writeln('Both = ', Area(5, 4) + Area(6, 3));
end.
```

Python

```
def area(length, width):
    return length * width

print("Room 1 =", area(5, 4))
print("Room 2 =", area(6, 3))
print("Both =", area(5, 4) + area(6, 3))
```

Sample run

```
Room 1 = 20
Room 2 = 18
Both = 38
```

💡 Built-in sub-programs you already use: writeln, readln, sqrt (Pascal); print, input, len, round (Python).

Key Words

Sub-program – A named part of a program that does one job.

Function – A sub-program that returns a value.

Procedure – A sub-program that does a job but returns no value.

Call – Using a sub-program by writing its name.

Parameter – A variable that receives a value sent into a sub-program.

Argument – The actual value sent when calling, e.g. 5 in area(5, 4).

Return value – The result a function gives back.

Reusability – Writing code once and using it many times.

Readability – How easy code is to read and understand.

Maintainability – How easy code is to fix and change.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

def – Python keyword to define a sub-program.

Built-in function – A function already provided by the language, e.g. len.

Sample Q&A

1. What is a sub-program?

> **Answer:** A named part of a program that does one job and can be called when needed.

2. Give three advantages of sub-programs.

> **Answer:** Reusability, readability and maintainability.

3. Difference between a function and a procedure?

> **Answer:** A function returns a value; a procedure does not.

4. What is GetGrade(66)?

> **Answer:** B.

5. What does area(6, 3) return?

> **Answer:** 18.

6. In area(5, 4), what are 5 and 4?

> **Answer:** Arguments (values for the parameters length and width).

MCQ

1. A named part of a program that does one job is a…

A. data type
B. comment
C. constant
D. sub-program

2. A sub-program that returns a value is a…

A. function
B. array
C. procedure
D. loop

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

3. In Pascal, a sub-program that returns no value is a…

A. record
B. const
C. procedure
D. function

4. “Write once, use many times” describes…

A. nesting
B. compiling
C. reusability
D. debugging

5. Fixing a job in one place only shows…

A. maintainability
B. iteration
C. random access
D. precedence

6. Clear names such as PrintLine improve…

A. readability
B. speed of the printer
C. the data type
D. memory size

7. GetGrade(82) returns…

A. S
B. B
C. C
D. A

8. GetGrade(47) returns…

A. C
B. S
C. W
D. B

9. area(5, 4) + area(6, 3) equals…

A. 18
B. 20
C. 9
D. 38

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

10. In def area(length, width):, length and width are…

A. reserved words
B. parameters
C. arrays
D. constants

11. The Python keyword that defines a sub-program is…

A. func
B. proc
C. def
D. sub

12. In Python, a function gives back its result with…

A. break
B. return
C. input
D. print

13. In Pascal, a function returns a value by…

A. assigning to the function name
B. using print
C. writing end.
D. using readln

14. ShowItem('Eggs', 55, 6) displays…

A. Eggs: Rs. 55
B. Eggs: Rs. 330
C. Eggs: Rs. 6
D. Eggs: Rs. 61

15. Which is a built-in Python function?

A. PrintLine
B. ShowItem
C. len
D. GetGrade

Structured Questions

1. State three advantages of using sub-programs. (3 marks) *(3 marks)*

> Reusability – write once, call many times; readability – easy to understand; maintainability – change in one place (1 each).

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

2. Distinguish between a value-returning and a non-value-returning sub-program with an example of each. (4 marks) *(4 marks)*

> Value-returning (function) gives back a value, e.g. GetGrade (2); non-value-returning (procedure) only does a task, e.g. PrintLine (2).

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

3. Write a function (Pascal or Python) that receives a mark and returns the O/L grade A, B, C, S or W. (5 marks) *(5 marks)*

Function heading with parameter (1); IF ≥ 75 A (1); ELSE IF ≥ 65 B, ≥ 50 C (1); ≥ 35 S else W (1); return / assign result (1).

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Value-returning sub-program (function): GetGrade

Pascal

```
program GradeFunction;
var
  k, m: integer;

function GetGrade(marks: integer): char;
begin
  if marks >= 75 then
    GetGrade := 'A'
  else if marks >= 65 then
    GetGrade := 'B'
  else if marks >= 50 then
    GetGrade := 'C'
  else if marks >= 35 then
    GetGrade := 'S'
  else
    GetGrade := 'W';
end;

begin
  for k := 1 to 3 do
  begin
    write('Enter mark: ');
    readln(m);
    writeln('Grade: ', GetGrade(m));
  end;
end.
```

Python

```
def get_grade(marks):
    if marks >= 75:
        return "A"
    elif marks >= 65:
        return "B"
    elif marks >= 50:
        return "C"
    elif marks >= 35:
        return "S"
    else:
        return "W"

for k in range(3):
    m = int(input("Enter mark: "))
    print("Grade:", get_grade(m))
```

Sample run

```
Enter mark: 82
Grade: A
Enter mark: 47
Grade: S
Enter mark: 66
Grade: B
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

4. def area(length, width): return length * width
What is displayed by print(area(5, 4) + area(6, 3))? Name the parameters and arguments. (3 marks) *(3 marks)*

> 38 (1); parameters: length, width (1); arguments: 5, 4, 6, 3 (1).

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

5. Write a procedure ShowItem that receives an item name, price and quantity and displays the item total. Show one call. (4 marks) *(4 marks)*

Procedure heading with three parameters (1); calculate price × qty (1); display (1); call e.g. ShowItem('Bread', 120, 2) (1).

Non-value-returning sub-programs (procedures): a shop bill

Pascal

```
program BillProcedures;

procedure PrintLine;
begin
  writeln('------------------------');
end;

procedure ShowItem(item: string; price, qty: integer);
begin
  writeln(item, ': Rs. ', price * qty);
end;

begin
  PrintLine;
  writeln('SAMPLE SHOP BILL');
  PrintLine;
  ShowItem('Bread', 120, 2);
  ShowItem('Milk packet', 280, 1);
  ShowItem('Eggs', 55, 6);
  PrintLine;
end.
```

Python

```
def print_line():
    print("-" * 24)

def show_item(item, price, qty):
    print(item + ": Rs.", price * qty)

print_line()
print("SAMPLE SHOP BILL")
print_line()
show_item("Bread", 120, 2)
show_item("Milk packet", 280, 1)
show_item("Eggs", 55, 6)
print_line()
```

Sample run

```
------------------------
SAMPLE SHOP BILL
------------------------
Bread: Rs. 240
Milk packet: Rs. 280
Eggs: Rs. 330
------------------------
```

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Summary

- Sub-program = named part doing one job.
- Benefits: reusability, readability, maintainability.
- Function returns a value.
- Procedure returns no value.
- Parameters receive; arguments are sent.
- Python: def … return.
- Pascal: FunctionName := value.
- Built-ins: print, len, writeln, sqrt.

MCQ answer key – 1.10

1. **D** – Sub-program.
2. **A** – Function.
3. **C** – Procedure.
4. **C** – Reusability.
5. **A** – Maintainability.
6. **A** – Readability.
7. **D** – 82 ≥ 75.
8. **B** – 35 ≤ 47 < 50.
9. **D** – 20 + 18.
10. **B** – Parameters.
11. **C** – def.
12. **B** – return.
13. **A** – GetGrade := 'A'.
14. **B** – 55 × 6.
15. **C** – Built-in.

© 2026 schoolict.net (Ruwan Mangala Suraweera). All rights reserved.

G.C.E. O/L Grade 11 ICT · Lesson 1

1.11 Programming Languages, Translators and Debugging

Syllabus 10.11. A computer only understands **0s and 1s**. People prefer words like print and if. In this part we see the different **kinds of programming languages**, how a **translator** (interpreter or compiler) changes our code into machine code, and how to **test and debug** a program.

1. Low-level and high-level languages

Type	What it looks like	Notes
Machine language	10110000 01100001	Only 0s and 1s. The CPU runs it directly. Very hard for people.
Assembly language	MOV AL, 61h	Short words (mnemonics). Needs an **assembler**. Depends on the CPU.
High-level language	print("Hello")	English-like, easy to learn, runs on many computers. Needs a compiler or interpreter. E.g. Python, Pascal, Java, C.

Machine and assembly languages are called **low-level** languages because they are close to the hardware.

2. Kinds of high-level languages

Procedural
Tell the computer **how** to do it, step by step. E.g. Pascal, C.

Declarative
Tell the computer **what** you want; it finds how. E.g. SQL, Prolog.

Structured
Built only from sequence, selection, iteration and sub-programs (no messy jumps). E.g. Pascal.

Object-oriented
Built from **objects** that hold data and actions together (e.g. a Student object). E.g. Java, C++, Python.

3. Programming and scripting languages

Scripting languages are usually interpreted and used to control other software or add actions to web pages – e.g. **JavaScript** (web pages), **PHP** (web servers), Python scripts. **Programming languages** such as C or Pascal are usually compiled into stand-alone programs.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

> NEW 2026 2026 – **Python** is now one of the most popular programming languages in the world. It is used for AI, data science, websites and school teaching, because its code is short and easy to read.

4. Translators: compiler and interpreter

	Compiler	Interpreter
How it translates	the whole program at once	one line at a time, then runs it
Output	an executable file (e.g. .exe)	no separate file
Errors	listed after compiling the whole program	stops at the first line with an error
Speed when running	faster	slower
Example	Free Pascal (FPC), C	Python

The code we write is the **source code**. The translated result is **machine code** (object code). An **assembler** translates assembly language.

5. Program errors

Error	Meaning	Example
Syntax error	Breaking the grammar rules of the language. The translator finds it.	prnt("Hi") / missing ;
Runtime error	The program stops while running.	dividing by zero
Logic error	The program runs but gives a **wrong answer**.	total / 2 instead of total / 3

6. Testing and debugging

Testing means running the program with chosen data and checking the output against the expected answer. **Debugging** means finding and fixing the errors (bugs). The program below should find the average of three marks, but it has a **logic error**.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

Logic error: dividing by the wrong number

Pascal

```
program AverageBug;
var
  m1, m2, m3, total: integer;
  avg: real;
begin
  write('Enter mark 1: ');
  readln(m1);
  write('Enter mark 2: ');
  readln(m2);
  write('Enter mark 3: ');
  readln(m3);
  total := m1 + m2 + m3;
  avg := total / 2;            { BUG: should be / 3 }
  writeln('Average = ', avg:0:2);
end.
```

Python

```
m1 = int(input("Enter mark 1: "))
m2 = int(input("Enter mark 2: "))
m3 = int(input("Enter mark 3: "))
total = m1 + m2 + m3
avg = total / 2               # BUG: should be / 3
print(f"Average = {avg:.2f}")
```

Sample run

```
Enter mark 1: 75
Enter mark 2: 68
Enter mark 3: 72
Average = 107.50
```

💡 Expected average of 75, 68 and 72 = 215 ÷ 3 = 71.67, but the program shows 107.50. Fix: divide by 3. Test with **normal** data (60), **boundary** data (0, 100, 35) and **invalid** data (−5, 150).

Ways to debug: read the error message, use a **trace table** (dry run), print variable values, and use the debugger in an IDE to run **step by step**.

7. IDEs and tools in 2026

An **IDE** (Integrated Development Environment) gives an editor, translator and debugger in one program.

- **IDLE** – comes free with Python; simple for beginners.
- **Thonny** – a beginner Python IDE that shows variables step by step.
- **VS Code** – a popular free editor for Python, Pascal and many languages.
- **Free Pascal / Lazarus** – for Pascal; **online compilers** run code in a web browser.
- **Scratch** – block coding: drag blocks instead of typing; a fun first step before Python.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

NEW 2026 2026 – **AI assistants** (such as chatbots inside VS Code) can explain errors and suggest code. Use them **responsibly**: try the problem yourself first, understand every line before using it, test the code, never submit AI work as your own in exams or assignments, and do not paste personal data into them.

Key Words

Machine language – Instructions in 0s and 1s that the CPU runs directly.

Assembly language – A low-level language using short mnemonics like MOV.

High-level language – An English-like language such as Python or Pascal.

Compiler – Translates the whole program at once into machine code.

Interpreter – Translates and runs one line at a time.

Assembler – Translates assembly language into machine code.

Source code – The program as written by the programmer.

Syntax error – A mistake in the grammar rules of the language.

Logic error – The program runs but gives a wrong result.

Runtime error – An error that stops the program while running.

Debugging – Finding and correcting errors in a program.

IDE – A program with an editor, translator and debugger together.

Sample Q&A

1. Why do high-level programs need a translator?

Answer: The CPU understands only machine code (0s and 1s).

2. Give one difference between a compiler and an interpreter.

Answer: A compiler translates the whole program at once; an interpreter translates one line at a time.

3. Procedural vs declarative?

Answer: Procedural says how (steps); declarative says what is wanted.

4. What kind of error is total / 2 instead of total / 3?

Answer: A logic error.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

5. Name two IDEs for Python.

> **Answer:** IDLE and Thonny (or VS Code).

6. Give one rule for using AI assistants responsibly.

> **Answer:** Understand and test the code; do not submit it as your own work.

MCQ

1. The language the CPU runs directly is…

A. Python
B. SQL
C. machine language
D. Pascal

2. MOV AL, 61h is written in…

A. assembly language
B. SQL
C. Python
D. HTML

3. Assembly language is translated by an…

A. editor
B. interpreter only
C. assembler
D. browser

4. Which is a high-level language?

A. Python
B. machine code
C. binary
D. assembly

5. A compiler translates…

A. only data
B. the whole program at once
C. one line at a time
D. only comments

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

6. Python is usually run by an…

A. interpreter
B. scanner
C. printer
D. assembler

7. SQL, which says what data is wanted, is…

A. machine code
B. declarative
C. assembly
D. procedural

8. A language built around objects that hold data and actions is…

A. machine
B. object-oriented
C. assembly
D. declarative

9. JavaScript, used to add actions to web pages, is a…

A. assembler
B. machine language
C. data type
D. scripting language

10. Writing prnt("Hi") in Python causes a…

A. hardware error
B. syntax error
C. logic error
D. no error

11. A program that runs but gives a wrong answer has a…

A. virus
B. syntax error
C. compiler
D. logic error

12. Dividing by zero while running causes a…

A. runtime error
B. syntax error
C. comment
D. logic error only

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

13. Marks 75, 68, 72 with total / 2 give Average = ...

A. 71.67
B. 72.50
C. 107.50
D. 215.00

14. Finding and fixing errors in a program is...

A. declaring
B. compiling
C. nesting
D. debugging

15. A responsible way to use an AI coding assistant is to...

A. submit its answers as your own work
B. paste your passwords into it
C. understand and test its code before using it
D. never test the code

Structured Questions

1. (i) Name two low-level languages. (1 mark)
(ii) Give two advantages of high-level languages. (2 marks) *(3 marks)*

> (i) Machine language, assembly language. (ii) English-like / easy to learn; runs on different computers; fewer errors (any two).

2. Compare a compiler and an interpreter under three headings. (3 marks) *(3 marks)*

> Translation: whole program vs line by line; output: executable file vs none; errors: listed together vs stops at the first; speed: faster vs slower (any three).

3. State the difference between (a) procedural and declarative languages (b) structured and object-oriented programming. (4 marks) *(4 marks)*

> (a) Procedural gives step-by-step how (Pascal); declarative states what is wanted (SQL) (2). (b) Structured uses sequence, selection, iteration and sub-programs; OOP builds programs from objects holding data and actions (2).

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

4. A program to find the average of three marks displays 107.50 for 75, 68, 72.
(i) What type of error is this? (1 mark)
(ii) What is the correct answer and how do you fix it? (2 marks)
(iii) Give two other types of error with an example. (2 marks) *(5 marks)*

(i) Logic error. (ii) 71.67; divide the total by 3. (iii) Syntax error – prnt("Hi"); runtime error – dividing by zero.

Logic error: dividing by the wrong number

Pascal

```
program AverageBug;
var
  m1, m2, m3, total: integer;
  avg: real;
begin
  write('Enter mark 1: ');
  readln(m1);
  write('Enter mark 2: ');
  readln(m2);
  write('Enter mark 3: ');
  readln(m3);
  total := m1 + m2 + m3;
  avg := total / 2;            { BUG: should be / 3 }
  writeln('Average = ', avg:0:2);
end.
```

Python

```
m1 = int(input("Enter mark 1: "))
m2 = int(input("Enter mark 2: "))
m3 = int(input("Enter mark 3: "))
total = m1 + m2 + m3
avg = total / 2               # BUG: should be / 3
print(f"Average = {avg:.2f}")
```

Sample run

```
Enter mark 1: 75
Enter mark 2: 68
Enter mark 3: 72
Average = 107.50
```

5. What is an IDE? Name two IDEs used for Python and state two rules for using AI assistants responsibly. (4 marks) *(4 marks)*

Editor, translator and debugger in one program (1); IDLE, Thonny or VS Code (1); try yourself first / understand and test the code / do not submit as your own / no personal data (2).

Summary

- Low-level: machine (0/1) and assembly.

© 2026 schoolict.net (Ruwan Mangala Suraweera) · All rights reserved

- High-level: English-like – Python, Pascal.
- Procedural = how; declarative = what.
- Structured vs object-oriented.
- Compiler: whole program; interpreter: line by line.
- Errors: syntax, runtime, logic.
- Test with normal, boundary, invalid data.
- IDEs: IDLE, Thonny, VS Code; use AI responsibly.

MCQ answer key – 1.11

1. **C** – 0s and 1s.
2. **A** – Mnemonics.
3. **C** – Assembler.
4. **A** – English-like.
5. **B** – Compiler.
6. **A** – Interpreter.
7. **B** – What, not how.
8. **B** – OOP.
9. **D** – Scripting.
10. **B** – Misspelt word.
11. **D** – Logic.
12. **A** – Runtime.
13. **C** – 215 ÷ 2.
14. **D** – Debugging.
15. **C** – Responsible use.

© 2026 schoolict.net (Ruwan Mangala Suraweera). All rights reserved.