Programming C/C++

Dr. Chayan Halder
Assistant Professor
Ramakrishna Mission Vivekananda Centenary College, Kolkata

Programming

What is Programming?

The communicative technique by which one can instruct a computing device to perform some task.

Why Programming?

Automated solution to the problems that can be solved with the aid of computing devices.

- Less time consuming
- No errors
- Repetitive

Programming

A typical programming task can be divided into two phases:

Problem solving phase

produce an ordered sequence of steps that describe solution of problem

this sequence of steps is called an ***algorithm***

Implementation phase

implement the program in some programming language

Problem Solving

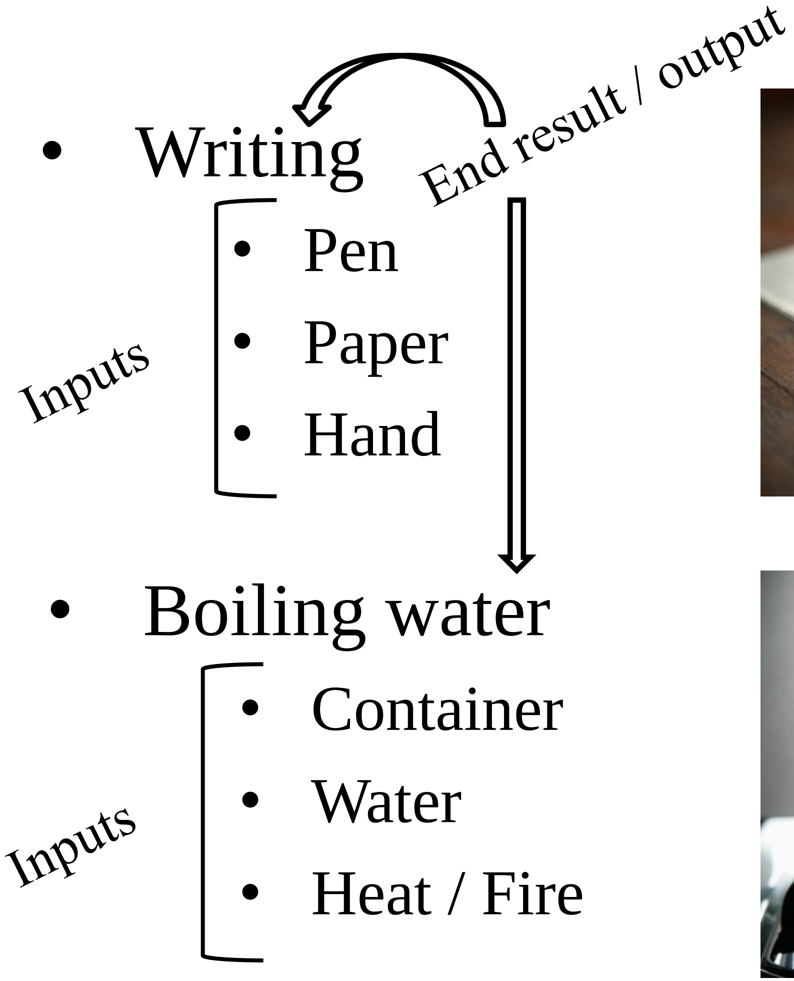
- Problem can be simple or complex

- Solutions of a problem can also be simple or complex.

- Solutions should have finite number of steps

- The steps should be sequential.

Problem Solving Example



- Writing
  - Pen
  - Paper
  - Hand
- Boiling water
  - Container
  - Water
  - Heat / Fire

End result / output

Inputs

Inputs

Algorithm

- An algorithm is a step by step recipe for solving an instance of a problem.
- Every single procedure that a computer performs is an outcome of some sort of algorithm.
- An algorithm is a precise procedure for solving a problem in finite number of steps.
- An algorithm states the actions to be executed and the order in which these actions are to be executed.

Algorithm

Definition

An algorithm is a well ordered collection of clear and simple instructions of definite and effectively computable operations that when executed produces a result and stops executing at some point in a finite amount of time rather than just going on and on infinitely.

Algorithm Example

- **Example 1:** Write an algorithm to determine a student’s final grade and indicate whether it is passing or failing. The final grade is calculated as the average of four marks.

Algorithm Example

Step1: Input 4 marks of a student

Step2: Calculate their average by summing and dividing by 4

Step3: If the average is below 50 then “FAIL” otherwise “PASS”

Algorithm Example

Algorithm

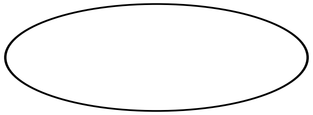
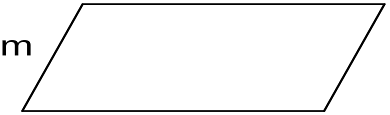
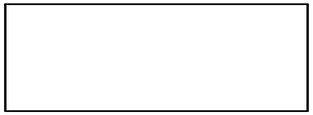
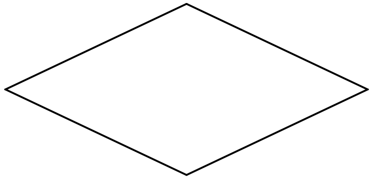
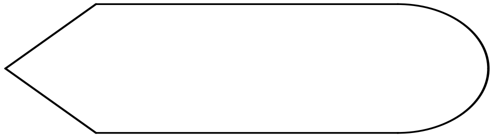
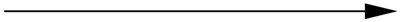
```
Step 1:   Input M1,M2,M3,M4
Step 2:   GRADE ← (M1+M2+M3+M4)/4
Step 3:   if (GRADE < 50) then
                      Print "FAIL"
              else
                      Print "PASS"
Step 4:   End
```

Flowchart

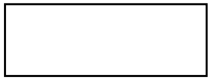
- Flowcharts is a graph used to depict or show a step by step solution using **symbols** which represent a task.
- The symbols used consist of geometrical shapes that are connected by **flow lines**.
- It is an alternative to pseudocoding; whereas a pseudocode description is verbal, a flowchart is graphical in nature.

Flowchart Symbols

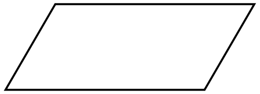
Basic

Name	Symbol	Use in Flowchart
Oval		Denotes the beginning or end of the program
Parallelogram		Denotes an input operation
Rectangle		Denotes a process to be carried out e.g. addition, subtraction, division etc.
Diamond		Denotes a decision (or branch) to be made. The program should continue along one of two routes. (e.g. IF/THEN/ELSE)
Hybrid		Denotes an output operation
Flow line		Denotes the direction of logic flow in the program

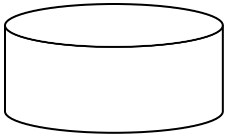
Flowchart Symbols (cont'd)



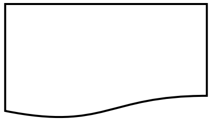
Process symbol - shows an instruction other than input, output or selection.



Input-output symbol - shows an input or an output operation.

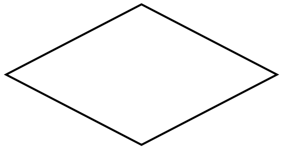


Disk storage I/O symbol - indicates input from or output to disk storage.

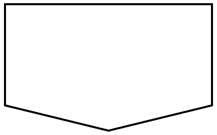


Printer output symbol - shows hardcopy printer output.

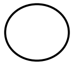
Flowchart Symbols (cont’d)



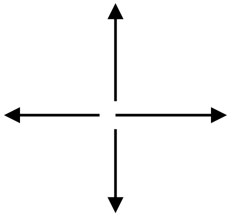
Selection symbol - shows a selection process for two-way selection.



Off-page connector - provides continuation of a logical path on another page.

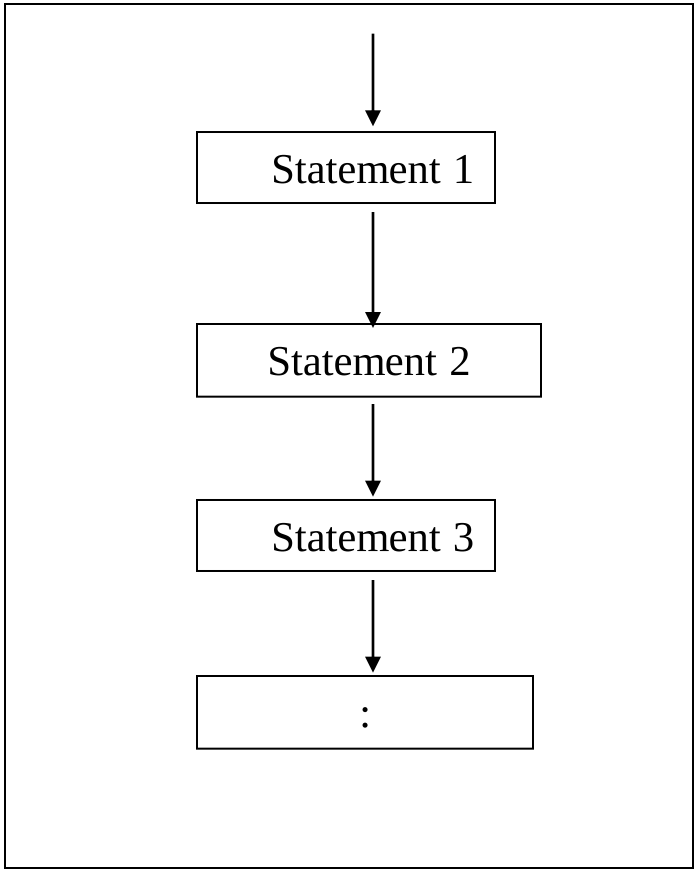


On-page connector - provides continuation of logical path at another point in the same page.

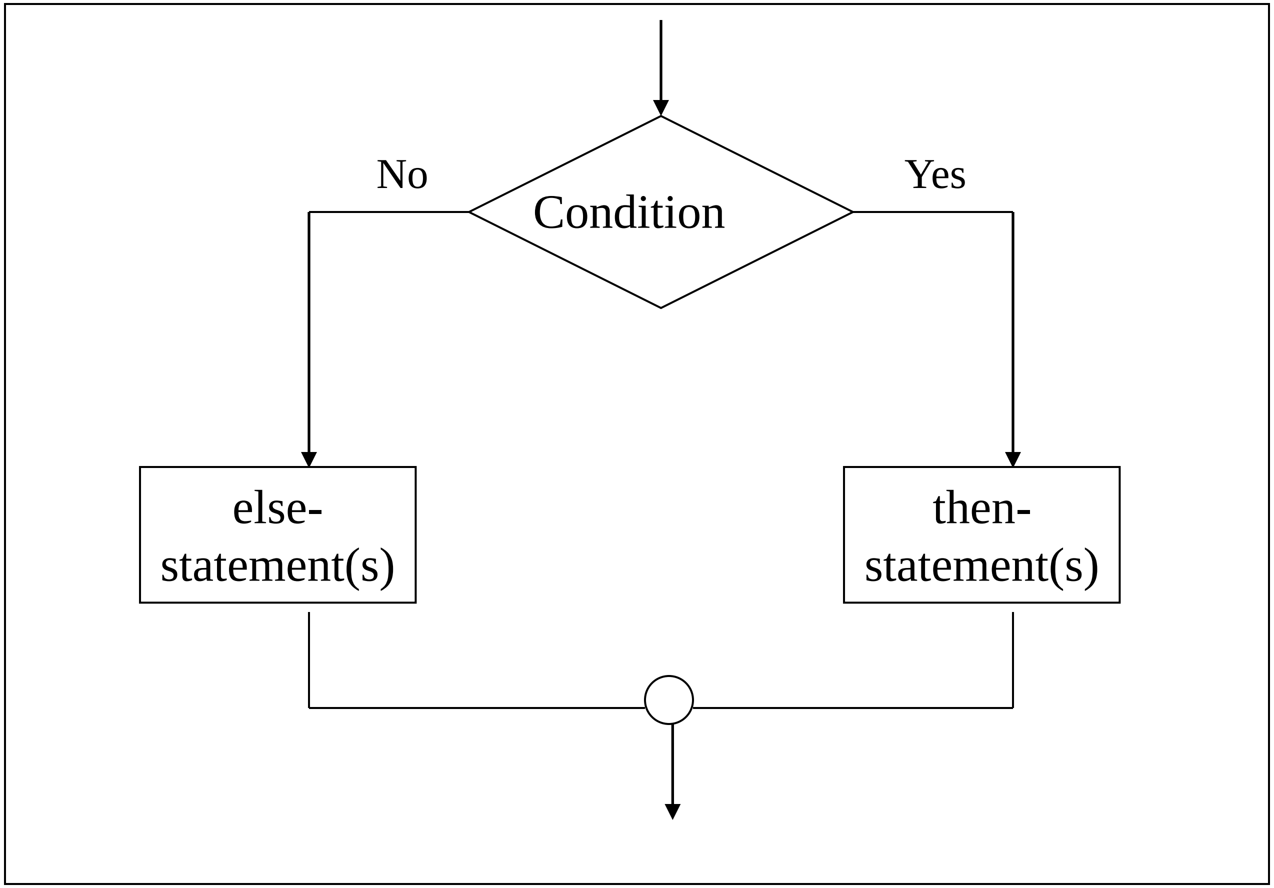


Flow lines - indicate the logical sequence of execution steps in the algorithm.

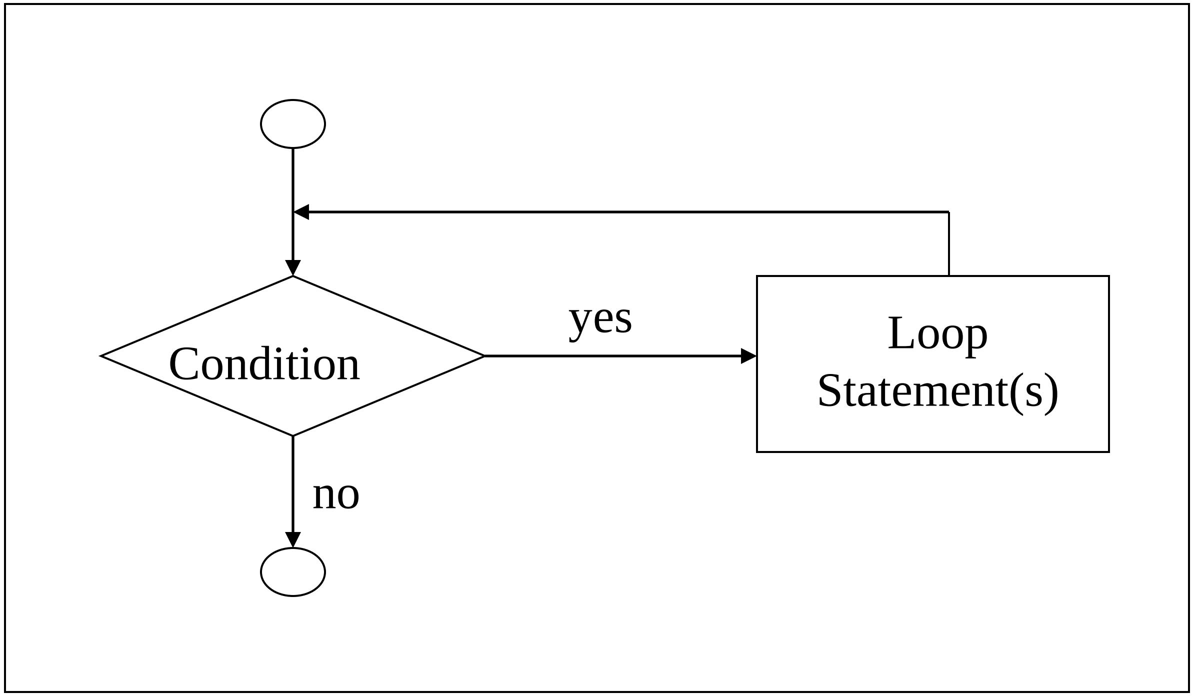
Flowchart – sequence control structure



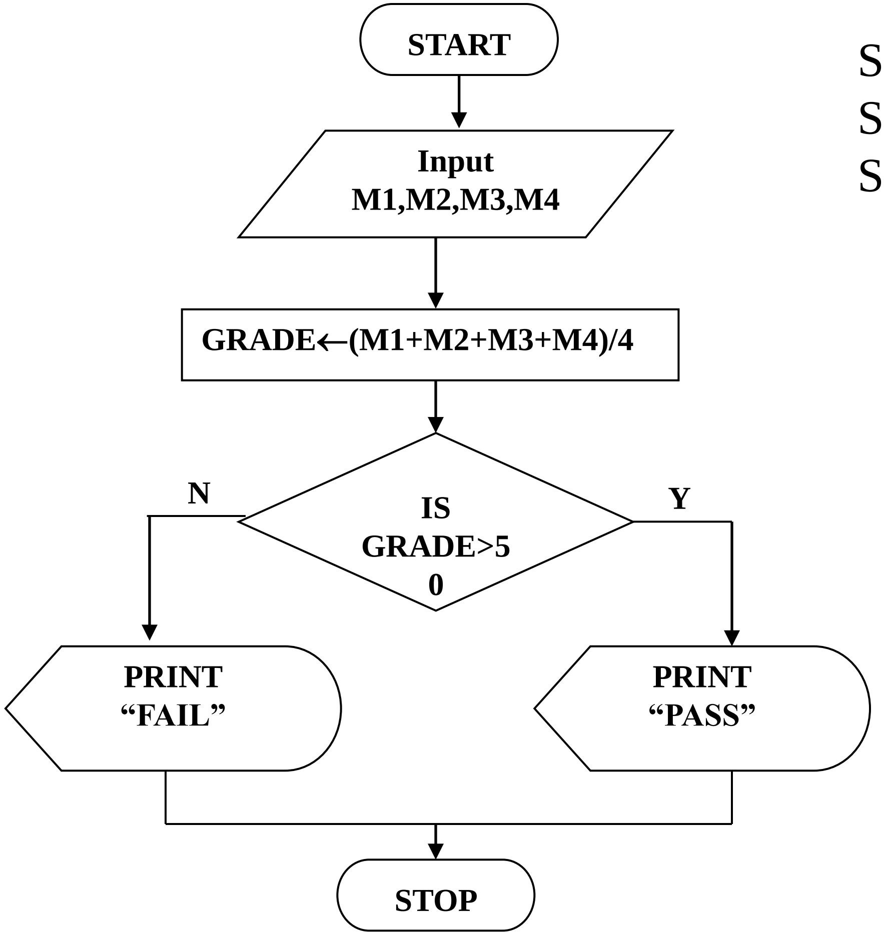
Flowchart – selection control structure



Flowchart – repetition control structure



Flowchart Example



```
Step 1:   Input M1,M2,M3,M4
Step 2:   GRADE ← (M1+M2+M3+M4)/4
Step 3:   if (GRADE <50) then
                    Print "FAIL"
          else
                    Print "PASS"
          end if
```

Flowchart Example (cont'd)

- Write an algorithm and draw a flowchart to convert the length in feet to centimeter.

Pseudocode:

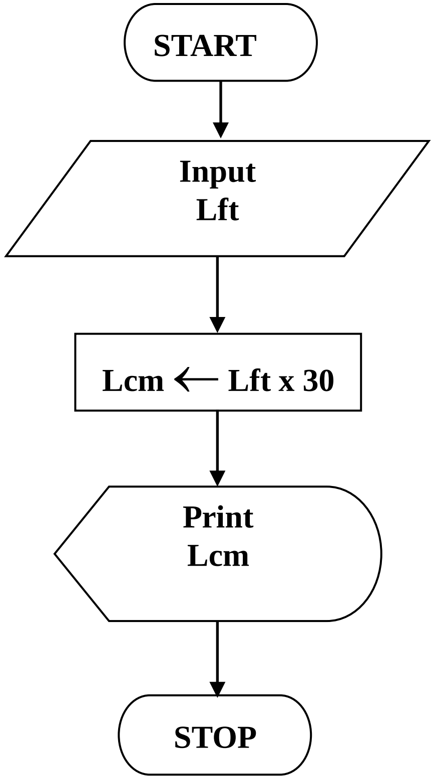
- *Input the length in feet (Lft)*
- *Calculate the length in cm (Lcm) by multiplying LFT with 30*
- *Print length in cm (LCM)*

Flowchart Example (cont’d)

Algorithm

- Step 1: Input Lft
- Step 2:   Lcm ← Lft x 30
- Step 3:   Print Lcm

Flowchart



Flowchart Example (cont’d)

Write an algorithm and draw a flowchart that will read the two sides of a rectangle and calculate its area.

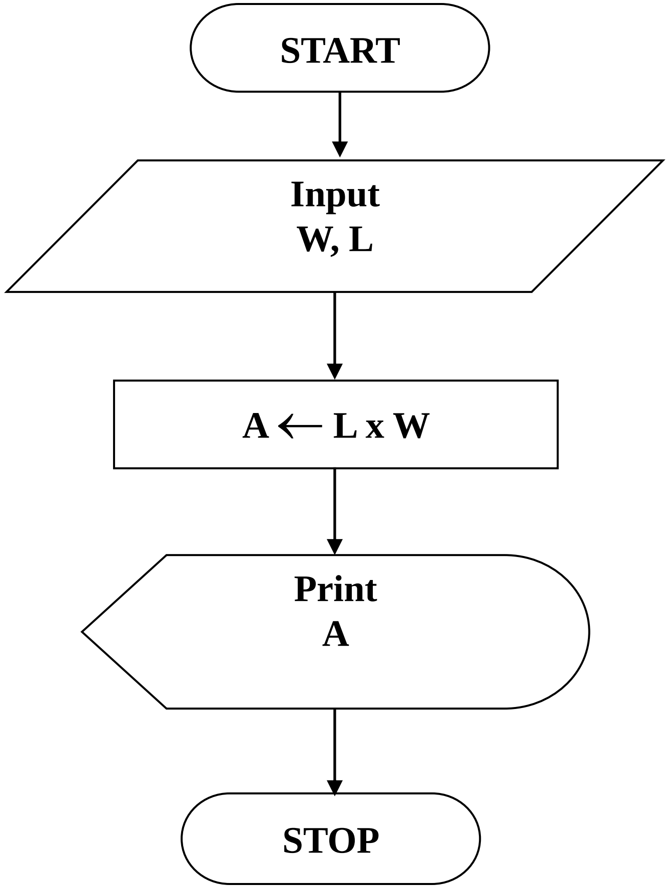
Pseudocode

- *Input the width (W) and Length (L) of a rectangle*
- *Calculate the area (A) by multiplying L with W*
- *Print A*

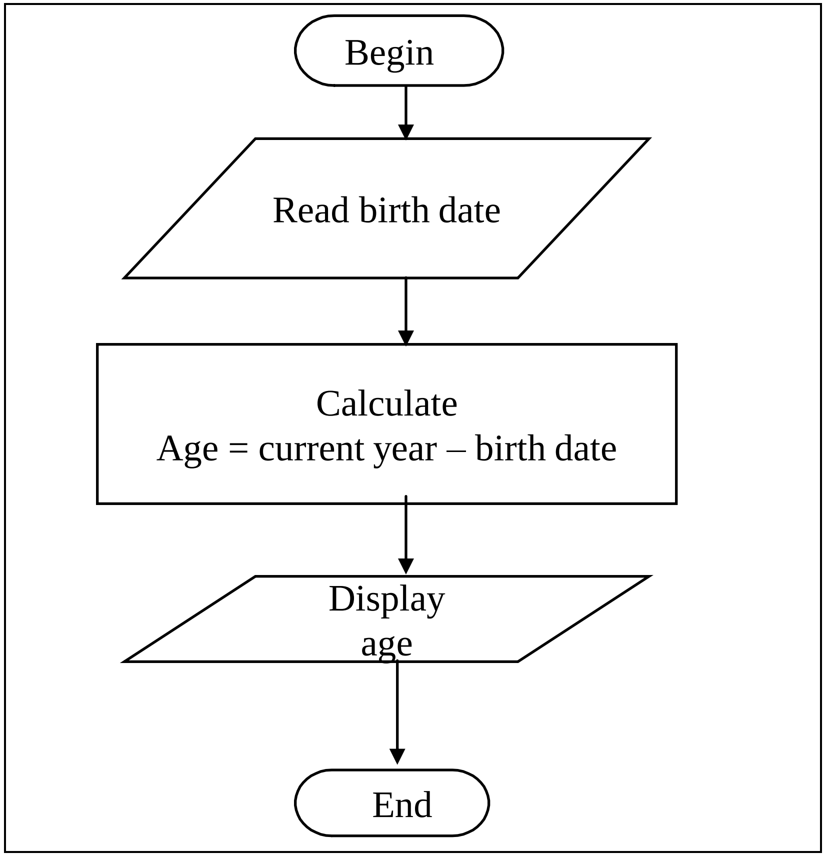
Flowchart Example (cont’d)

Algorithm

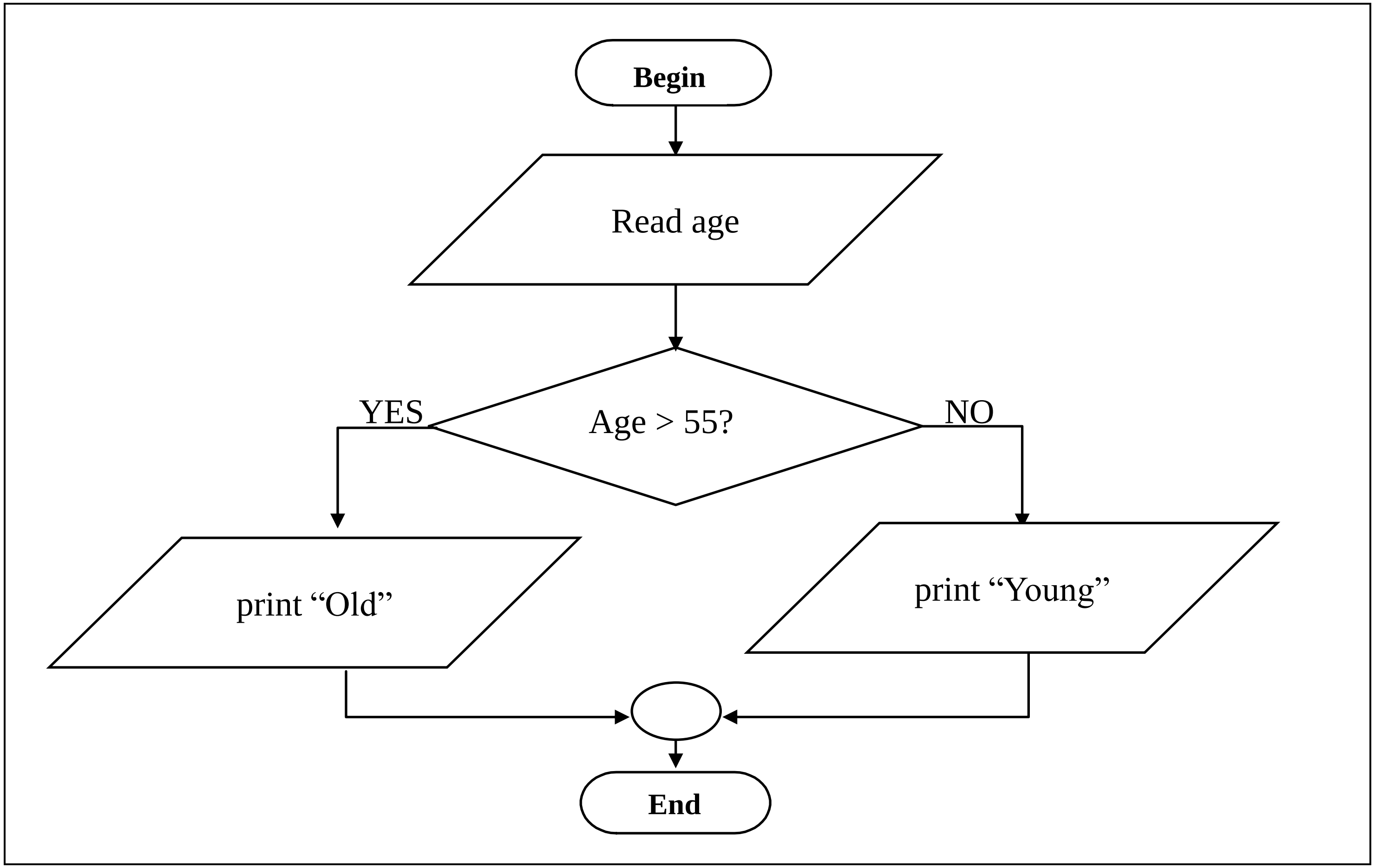
- Step 1: Input W,L
- Step 2: A ← L x W
- Step 3: Print A



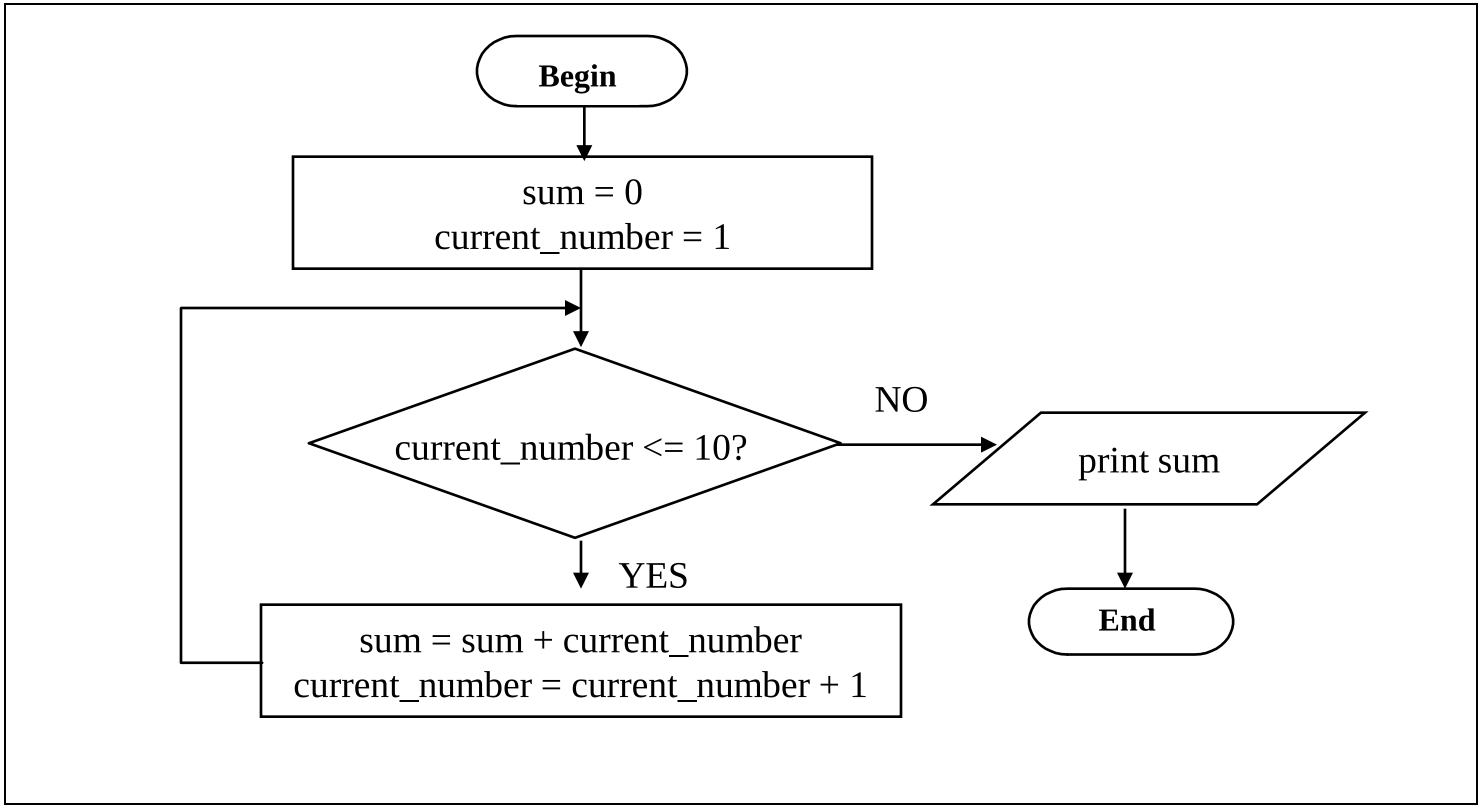
Flowchart Example (cont’d)



Flowchart Example (cont’d)



Flowchart Example (cont’d)



Begin

sum = 0
current_number = 1

current_number <= 10?

NO

print sum

End

YES

sum = sum + current_number
current_number = current_number + 1