Module 1
Introduction to Structured Programming methodology

1.1

Problem solving skill development

What is Programming?

- Giving instructions to a computer to perform a task.
- Why do we program?
    - To automate tasks, solve problems.

Introduction to Problem Solving Methodology

- It's a systematic approach
- Steps:

Define Problem -> Analyze -> **Design** -> Implement -> Test -> Maintain

What is an Algorithm?

- A step-by-step procedure to solve a problem.
- Properties: Finite, unambiguous, effective, input, output.

Algorithm Example
Sum of Two Numbers

- **Read two numbers, add them, display result.**
- Algorithm:
  - Start
  - Read num1
  - Read num2
  - sum = num1 + num2
  - Display sum
  - Stop

What is a Flowchart?

- A graphical representation of an algorithm
- Uses standard symbols to depict flow.
- Visualizing logic and control flow.

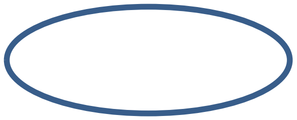
Flowcharts

- 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.

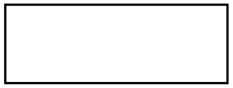
Flowchart

- Shows logic of an algorithm
- Emphasizes individual steps and their interconnections
- A graphical representation of the sequence of operations in an information system or program.
- Program flowcharts show the sequence of instructions in a single program or subroutine. Different symbols are used to draw each type of flowchart.

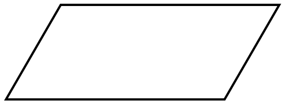
Flowchart Symbols



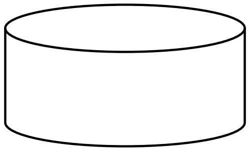
Terminal symbol - indicates the beginning and end points of an algorithm.



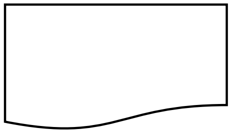
Process symbol - shows an instruction other
than input, output
or selection.



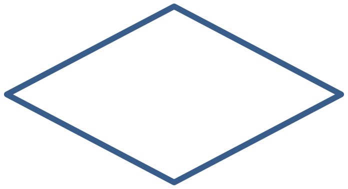
Input-output symbol - shows an input or an output operation.



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



Printer output symbol - shows hardcopy printer output.



Decision symbol – Program should continue along one of two routes
(If –then -else)

Steps in Problem Solving

- First produce a general algorithm (one can use ***pseudocode***)
- Refine the algorithm successively to get step by step detailed ***algorithm*** that is very close to a computer language.
- ***Pseudocode*** is an artificial and informal language that helps programmers develop algorithms. Pseudocode is very similar to everyday English.

Pseudocode & Algorithm

- **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.

Pseudocode & Algorithm

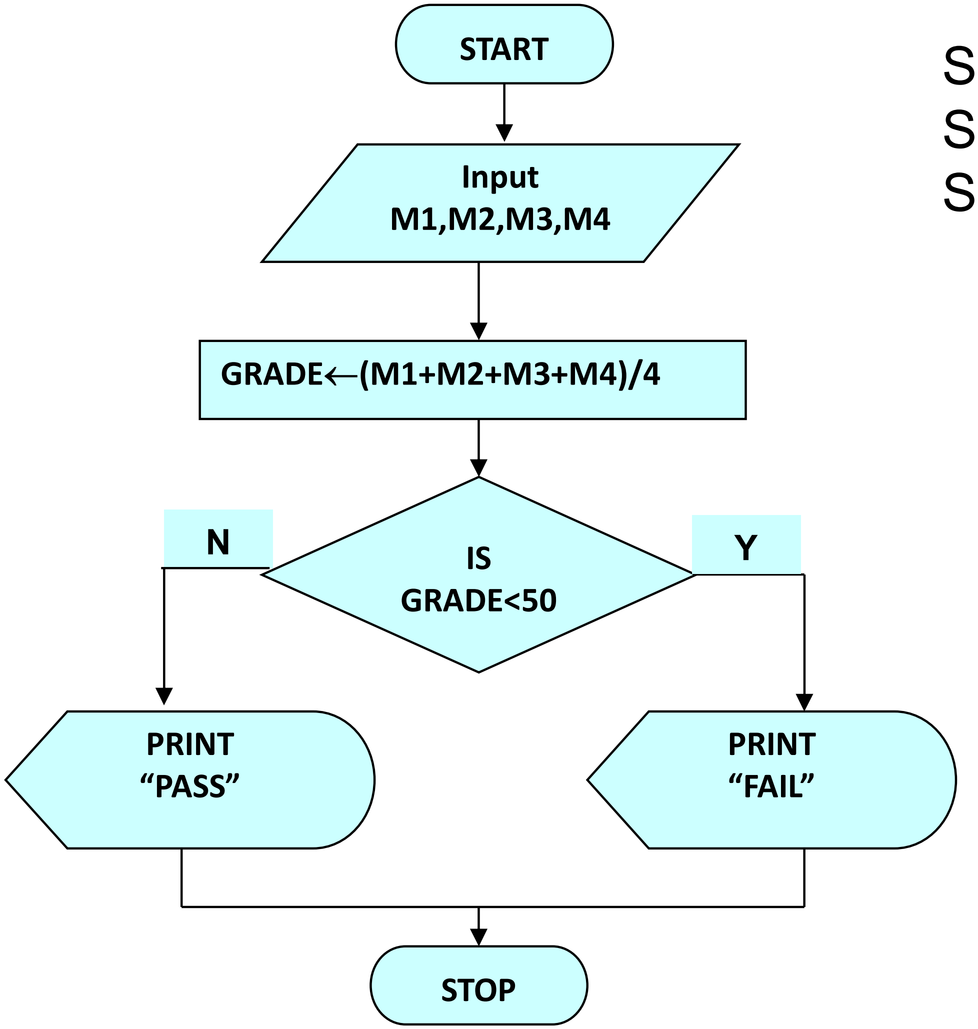
Pseudocode:

```
• Input a set of 4 marks
• Calculate their average by summing and dividing by 4
• if average is below 50
          Print "FAIL"
  else
          Print "PASS"
```

Pseudocode & Algorithm

- Detailed 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”
           endif
```

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”
          endif
```

Example 2

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

Pseudocode:

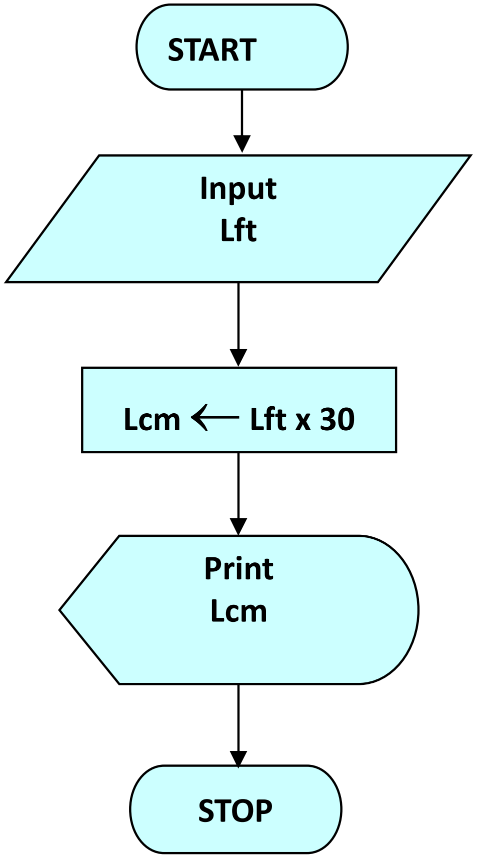
- *Input the length in feet (Lft)*
- *Calculate the length in cm (Lcm) by multiplying LFT with 30*
- *Print length in cm (LCM)*

Example 2

Algorithm

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

Flowchart



Example 3

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

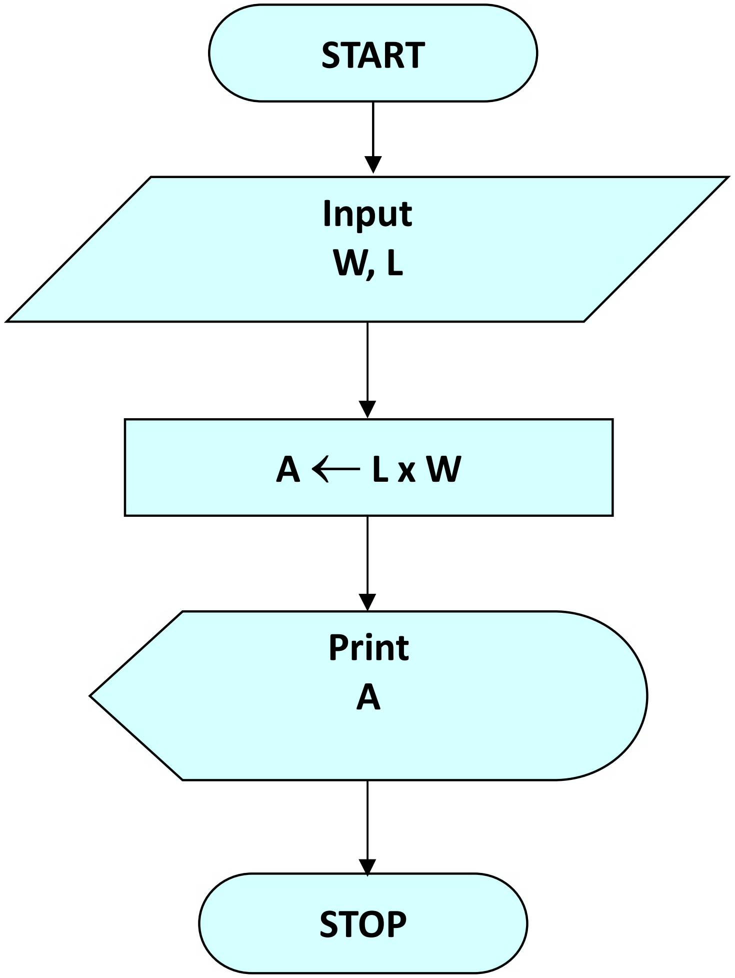
Pseudocode

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

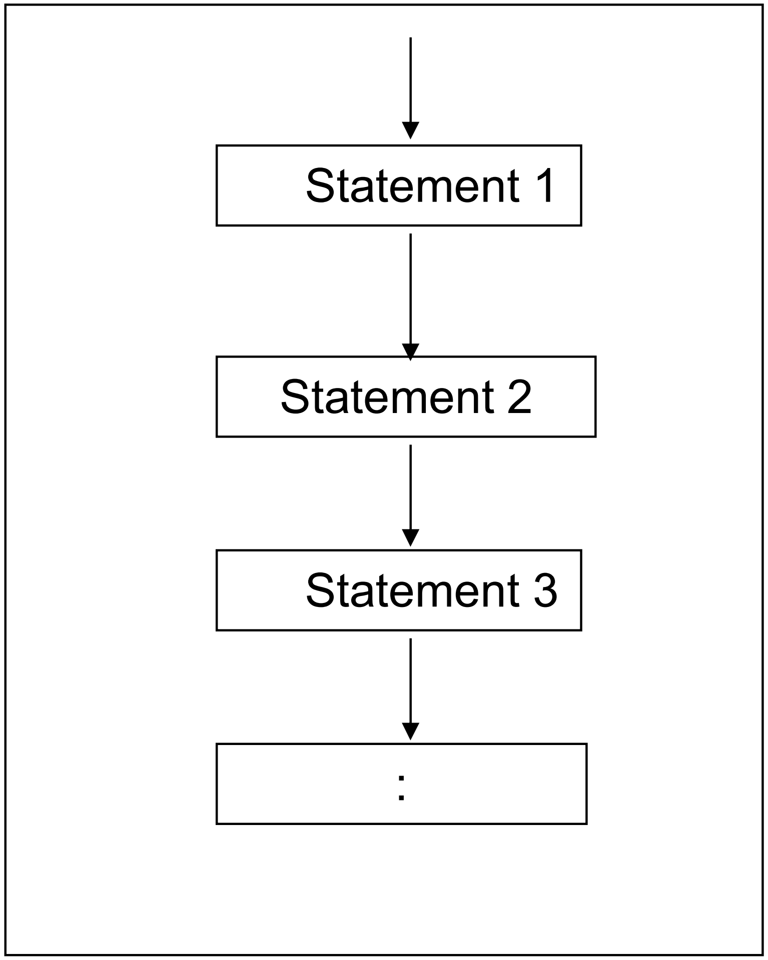
Example 3

Algorithm

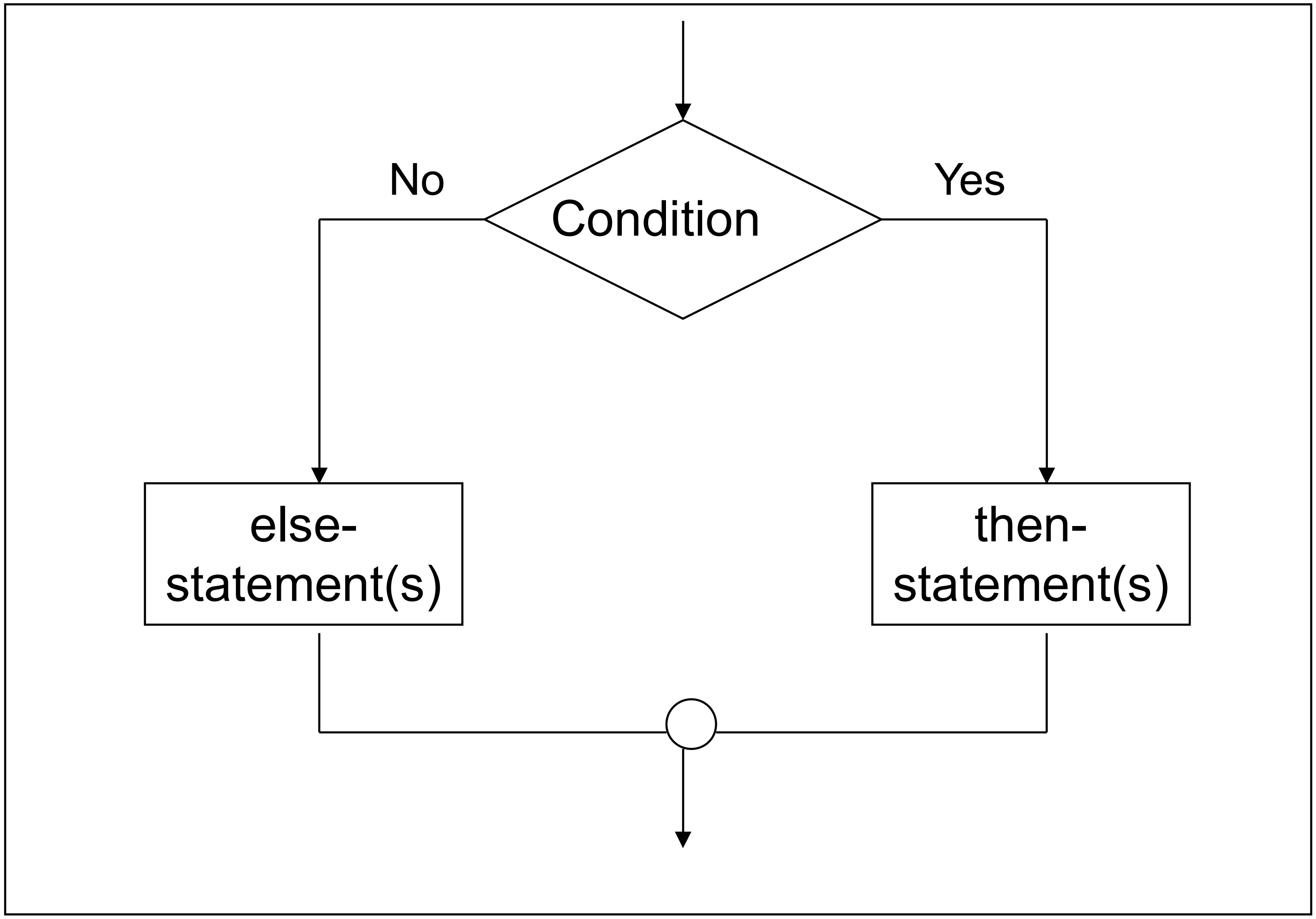
- Step 1: Input W,L
- Step 2: A ← L x W
- Step 3: Print A



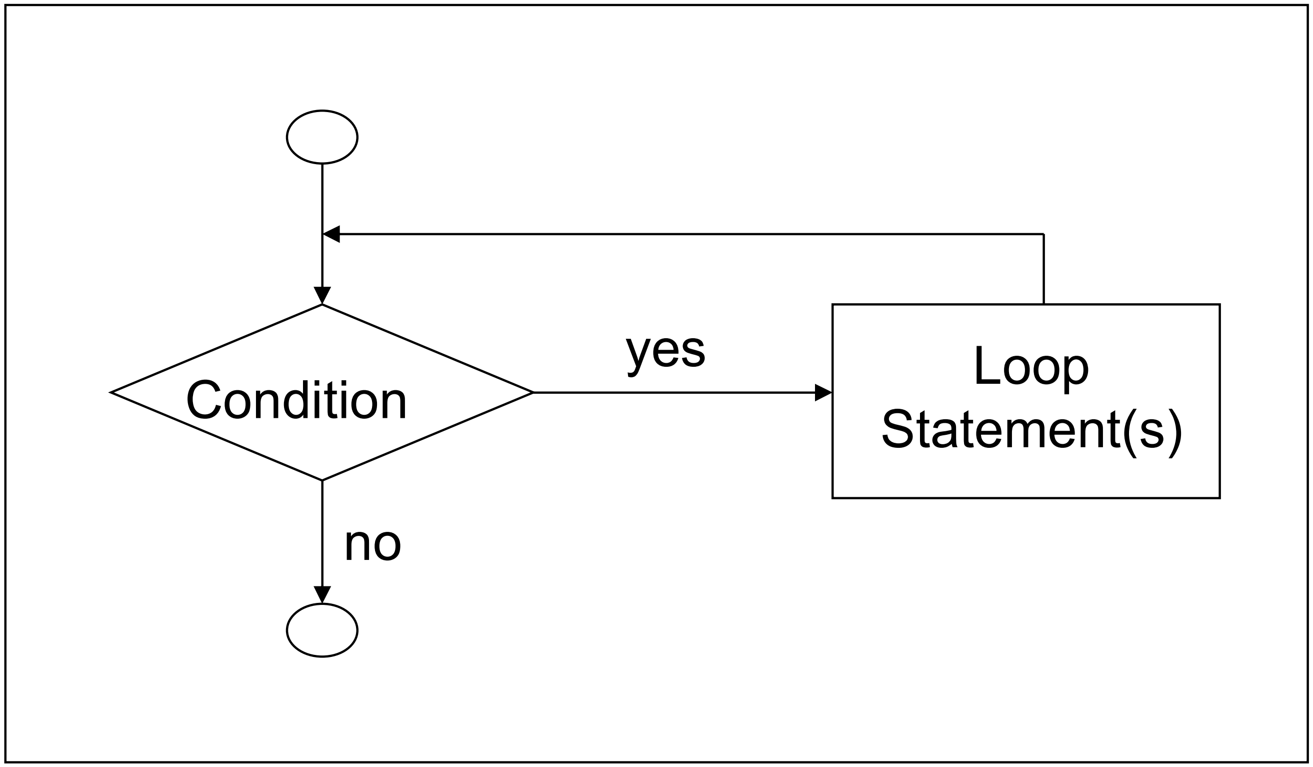
Flowchart – sequence control structure



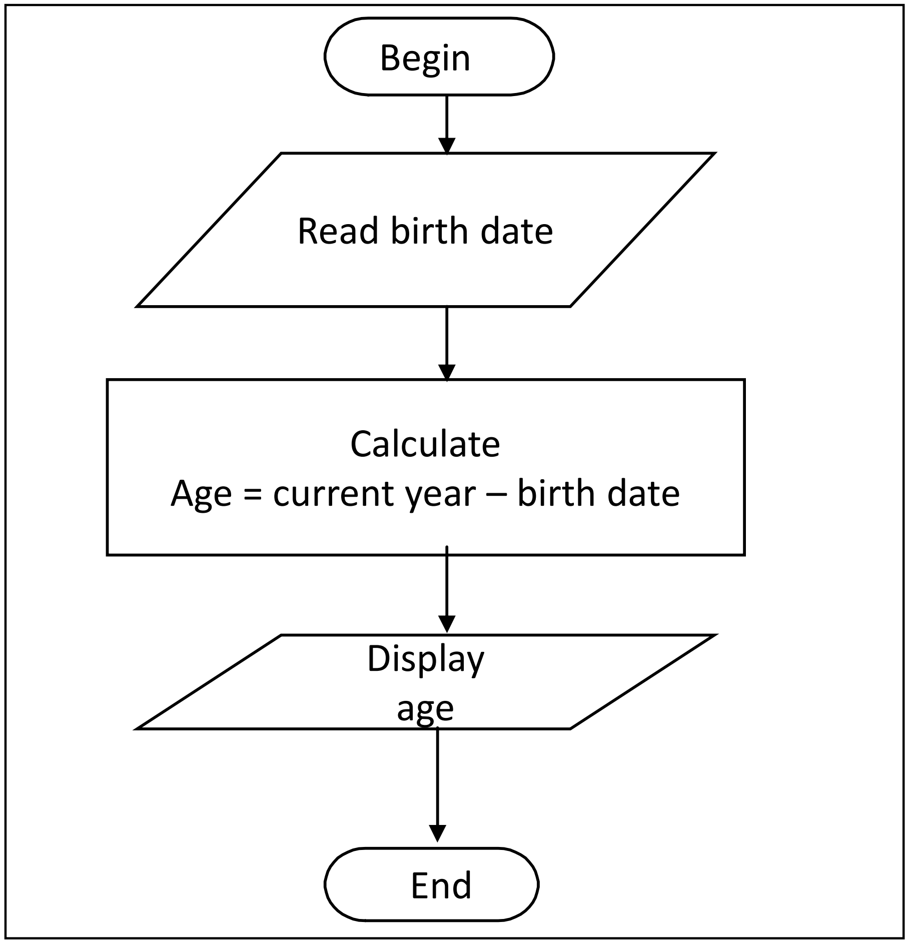
Flowchart – selection control structure



Flowchart – repetition control structure



Flowchart – example



Begin

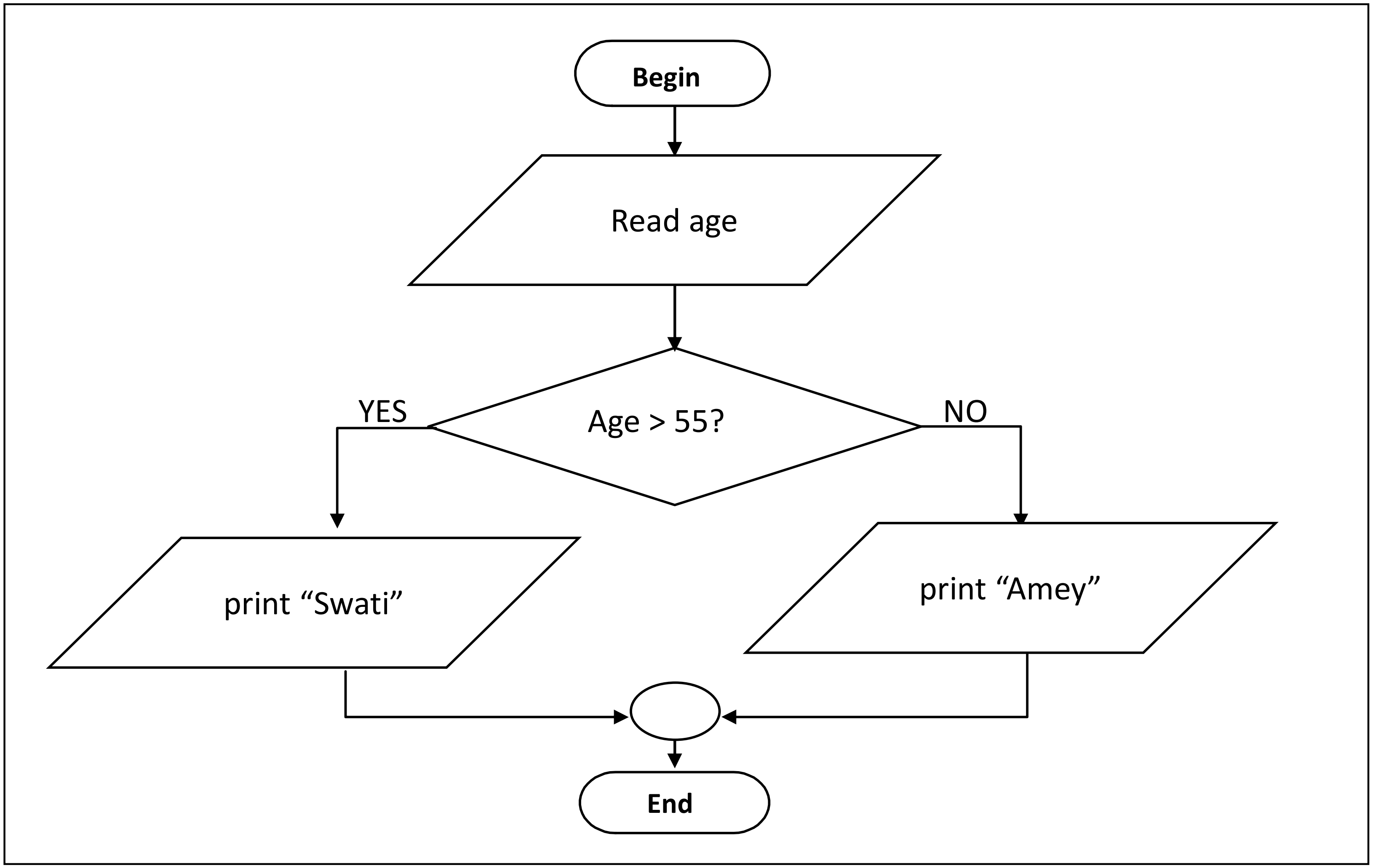
Read birth date

Calculate
Age = current year – birth date

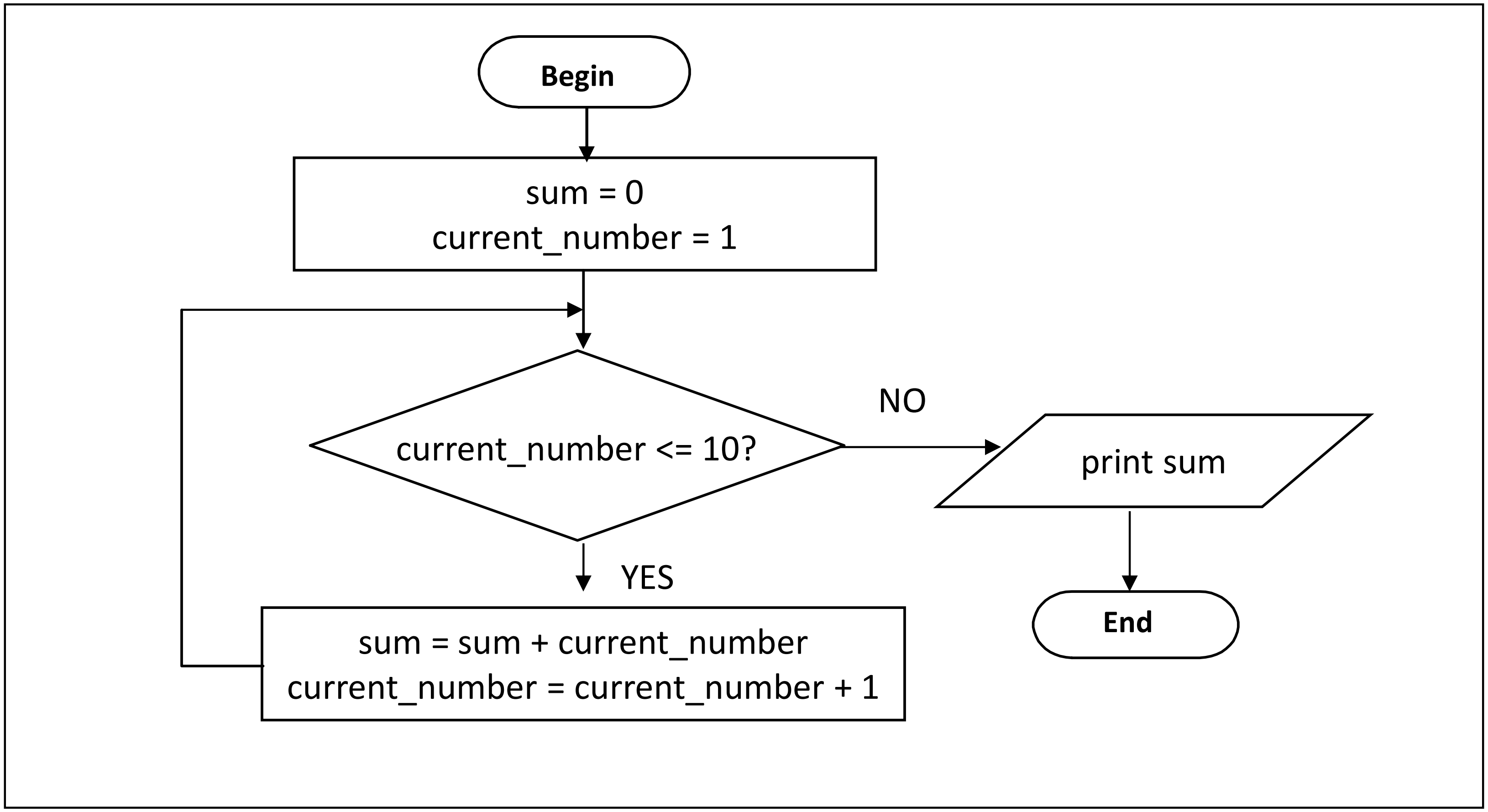
Display
age

End

Flowchart – example 2



Flowchart – example



Example

- Write an algorithm and draw a flowchart that will calculate the roots of a quadratic equation $ax^2 + bx + c = 0$
- Hint: **d** = sqrt $(b^2 - 4ac)$,

  and the roots are:

  ***x*1** = $(-b + d)/2a$ and ***x*2** = $(-b - d)/2a$

Exercises: Algorithm & Flowchart

1.) Create an algorithm and a flowchart that will accept/read two numbers and then display the bigger number.

Exercises: Algorithm & Flowchart

2.) Create an algorithm and a flowchart that will compute the area of a circle.

Exercises: Algorithm & Flowchart

3.) Create an algorithm and a flowchart that will compute the sum of two numbers. If the sum is below or equal to twenty, two numbers will be entered again. If the sum is above 20, it will display the sum.

Lab Activity: Algorithm & Flowchart

4) Create an algorithm and a flowchart that will output the largest number among the three numbers.

Assignment 1

1. Create an algorithm and a flowchart that will output for GCD (Greatest Common Divisor).

The Euclidean Algorithm for finding GCD(A,B) is as follows:
If A = 0 then GCD(A,B)=B, since the GCD(0,B)=B, and we can stop.
If B = 0 then GCD(A,B)=A, since the GCD(A,0)=A, and we can stop.
Write A in quotient remainder form (A = B·Q + R)
Find GCD(B,R) using the Euclidean Algorithm since GCD(A,B) = GCD(B,R)

Example:

Find the GCD of 270 and 192

A=270, B=192 A ≠0 B ≠0

Use long division to find that 270/192 = 1 with a remainder of 78. We can write this as: **270 = 192 * 1 +78**

Find GCD(192,78), since GCD(270,192)=GCD(192,78)

A=192, B=78

A ≠0 B ≠0

Use long division to find that 192/78 = 2 with a remainder of 36. We can write this as:

192 = 78 * 2 + 36

Find GCD(78,36), since GCD(192,78)=GCD(78,36)

A=78, B=36

A ≠0   B ≠0

Use long division to find that 78/36 = 2 with a remainder of 6. We can write this as:

78 = 36 * 2 + 6   Find GCD(36,6), since GCD(78,36)=GCD(36,6)

A=36, B=6

A ≠0   B ≠0

Use long division to find that 36/6 = 6 with a remainder of 0. We can write this as:

36 = 6 * 6 + 0   Find GCD(6,0), since GCD(36,6)=GCD(6,0)

A=6, B=0   A ≠0

B =0, GCD(6,0)=6

So we have shown:

GCD(270,192) = GCD(192,78) = GCD(78,36) = GCD(36,6) = GCD(6,0) = 6

GCD(270,192) = 6

Other way_to find GCD

Step 1: Initialise a variable gcd to 1. This variable will store the greatest common divisor of the input numbers n1 and n2.

Step 2: Iterate from 1 to the minimum of n1 and n2.

We start from 1 because the GCD of any two numbers is at least 1, and it cannot be greater than the smaller of the two numbers.

Step 3: At each iteration, if i is a common factor of both n1 and n2 update the gcd variable to i. We keep updating gcd as long as we find common factors.

Step 4: After the iteration, the gcd variable will store the greatest common divisor of n1 and n2. Return this value as the output of the function

Assignment 1

1. Create an algorithm and a flowchart that will output for GCD (Greatest Common Divisor).

Start

Input: two non-negative integers, a and b.

Loop: while b is not equal to 0:

a. Calculate the remainder r when a is divided by b (r = a % b).

b. Assign the value of b to a (a = b).

c. Assign the value of r to b (b = r).

Output: a (which now holds the GCD).

End

Assignment 1

1. Create an algorithm and a flowchart that will output for GCD (Greatest Common Divisor).
2. Create an algorithm and a flowchart that will output the factorial of a given number.
3. Create an algorithm and a flowchart that will output the Fibonacci series up to a given number.
4. Create an algorithm and a flowchart that will output all the prime numbers between 2 numbers.