Algorithms and Flowcharts for Programming CFD

Dr. Ugur GUVEN

Importance of Algorithms for Programming CFD

- If you want to be successful in your programming, it is essential for you to first develop an algorithm before proceeding with your program
- An algorithm must contain the following:

  1) What data do you need inputted?

  2) What is the function that you are going to evaluate?

  3) Do You Have any special conditions for evaluation? If Yes Then What?

  4) Calculation of the Function using the relevant numerical method and the Given Boundary Conditions of the Flow

  5) Printing of the Calculated Results

Example of an Algorithm

- Here is an example of an algorithm of a simple program that converts Fahrenheit into Celsius

1. Read Fahrenheit temperature
2. Calculate and write Celsius temperature
3. Stop.

Step 1 is pretty straightforward, but step 2 needs elaborating, so the second level plan could be something like this:

1. Input Fahrenheit temperature (F)
2. Calculate Celsius temperature (C):
   2.1   Subtract 32 from F and multiply by 5/9
3. Output the value of C
4. Stop.

Flowcharts

- Flowchart is the second step after writing the algorithm.

- You have to use special flowchart symbols that allow you to state the necessary operations that need to be executed by the computer.

- Flowcharts are unbounded by the programming language used. They will depict the logical flow of the program.

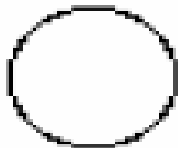# Flowchart Symbols for Programs

Start/Stop

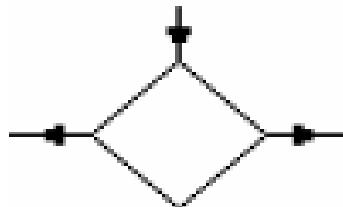Input or assignment of data

Processing (e.g. calculation of formulae)

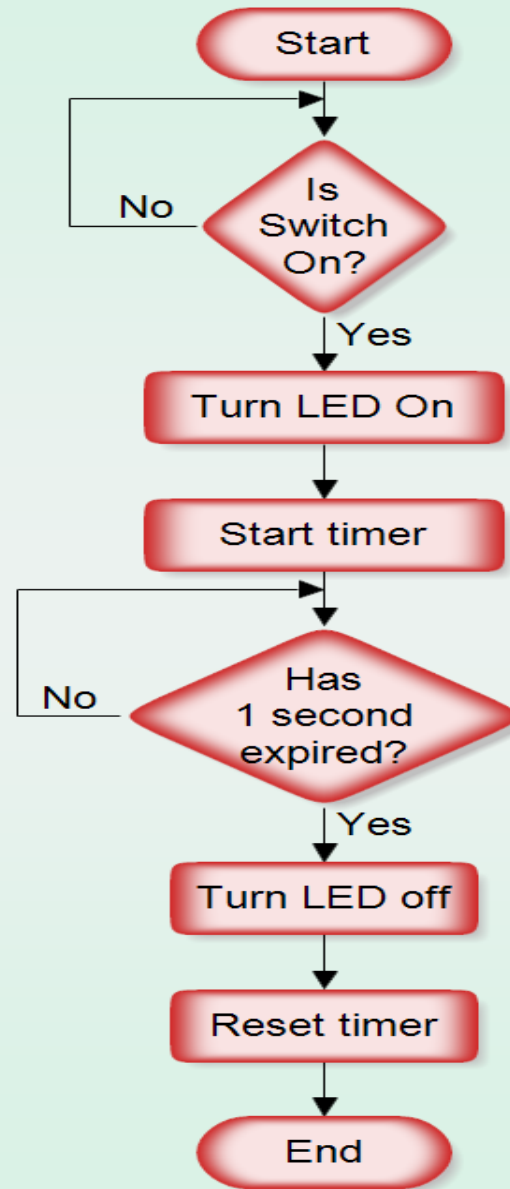Header for loop structure
Decision

End of loop structure

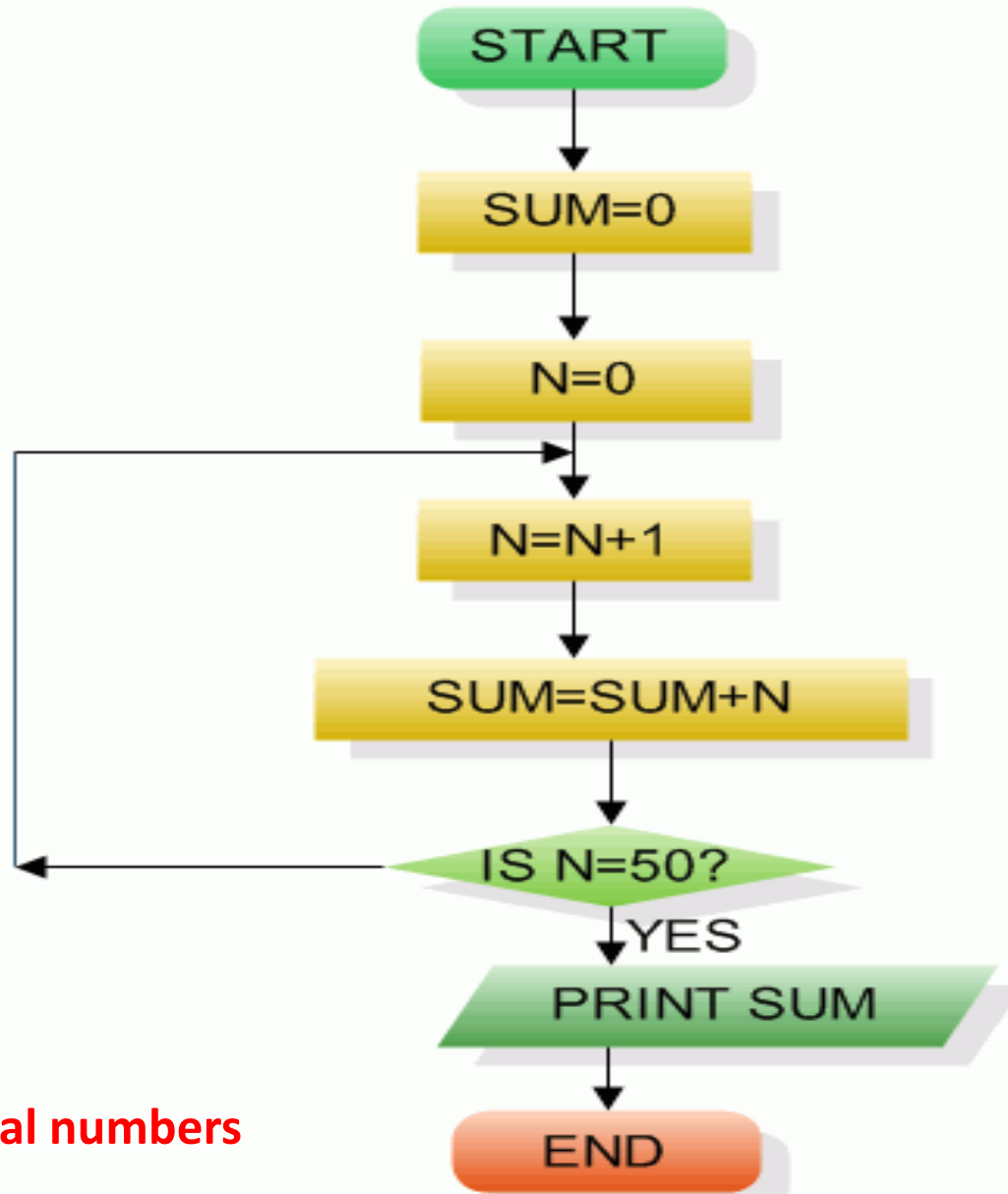Decision (two- or three-way branch)

Output

Sample Flowchart

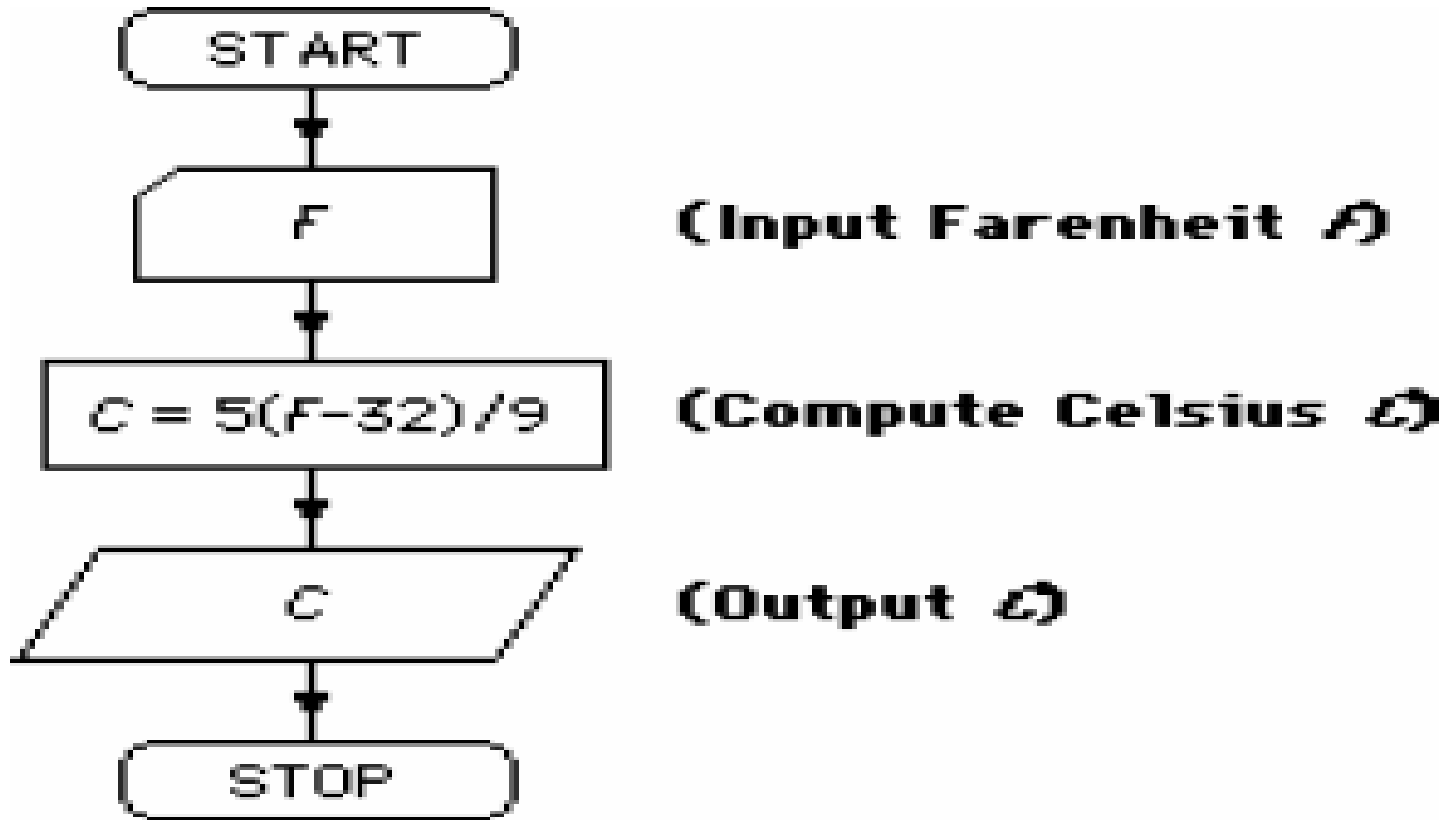Flowchart of a One Second Response to a Switch

Sample Flowchart

Sum of first 50 natural numbers

Sample Flowchart

- Here is the sample flowchart for the Fahrenheit to Celsius conversion program:

```
        ┌──────────────┐
        (    START     )
        └──────────────┘
               │
         ┌───────────┐
        /     F      /        (Input Farenheit F)
       └────────────┘
               │
      ┌──────────────────┐
      │ C = 5(F-32)/9    │    (Compute Celsius C)
      └──────────────────┘
               │
         ┌───────────┐
        /     C      /        (Output C)
       └────────────┘
               │
        ┌──────────────┐
        (    STOP      )
        └──────────────┘
```

Sample Flowchart

- Here is the sample algorithm for Newton's Method for Calculating Square Roots:
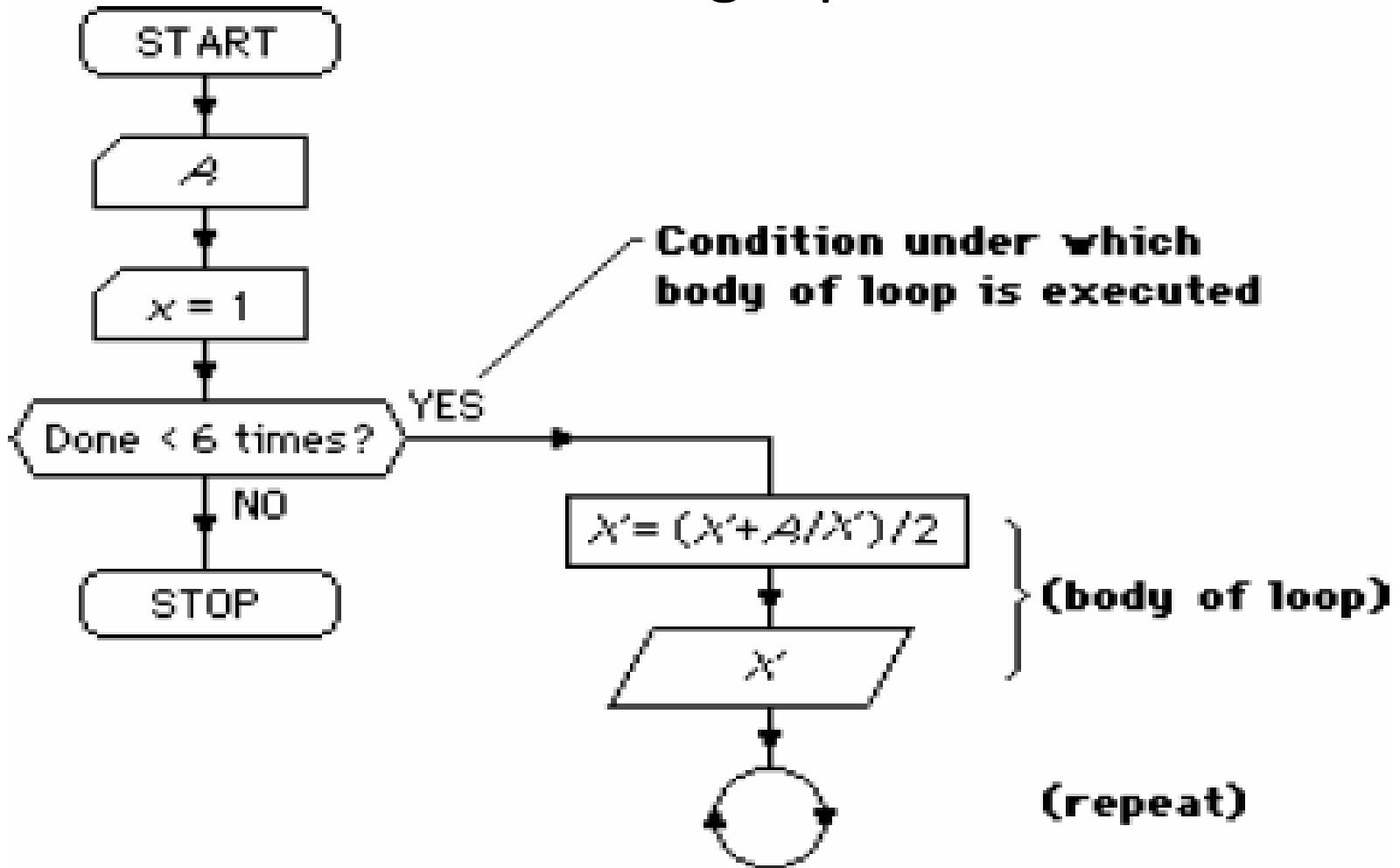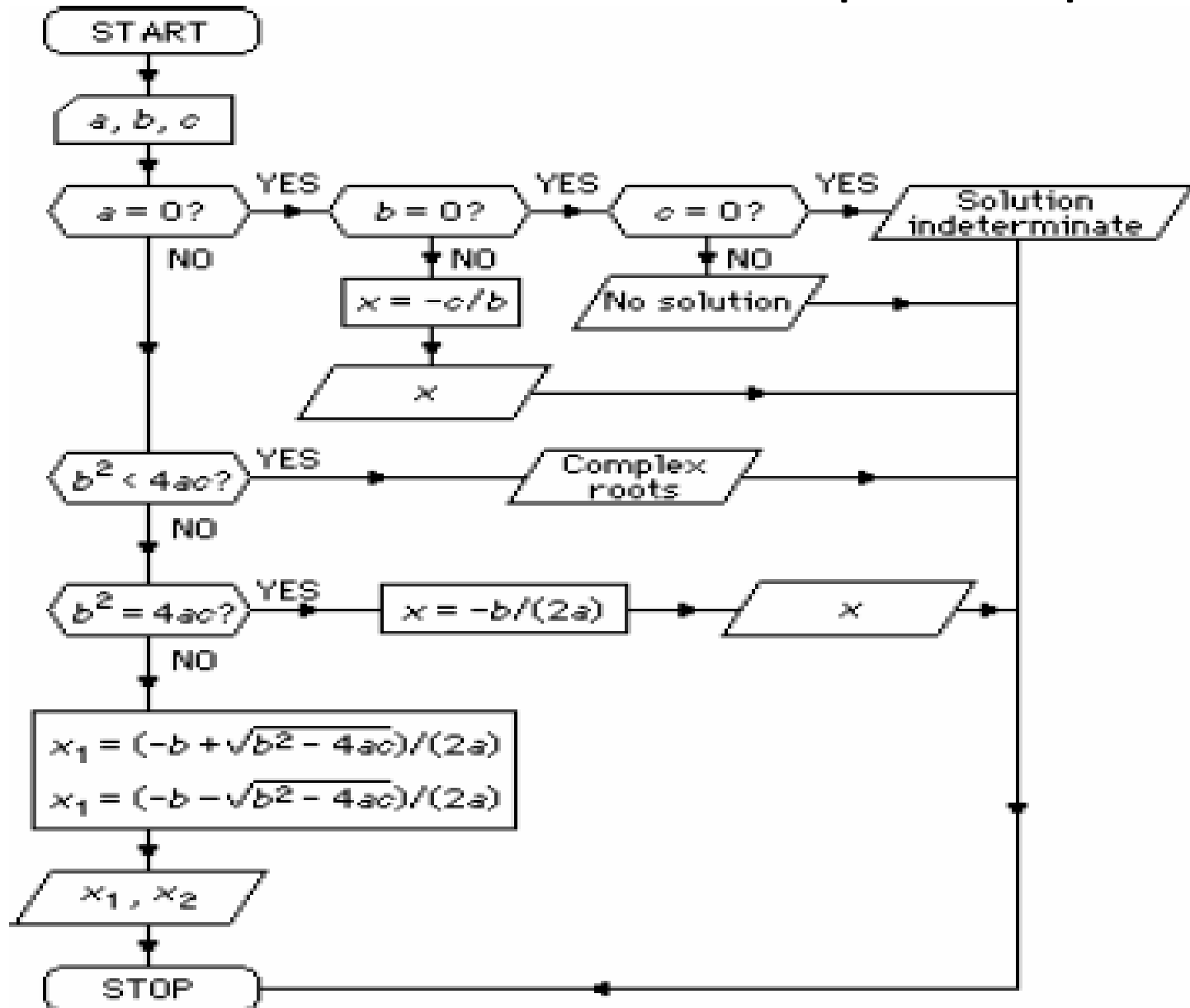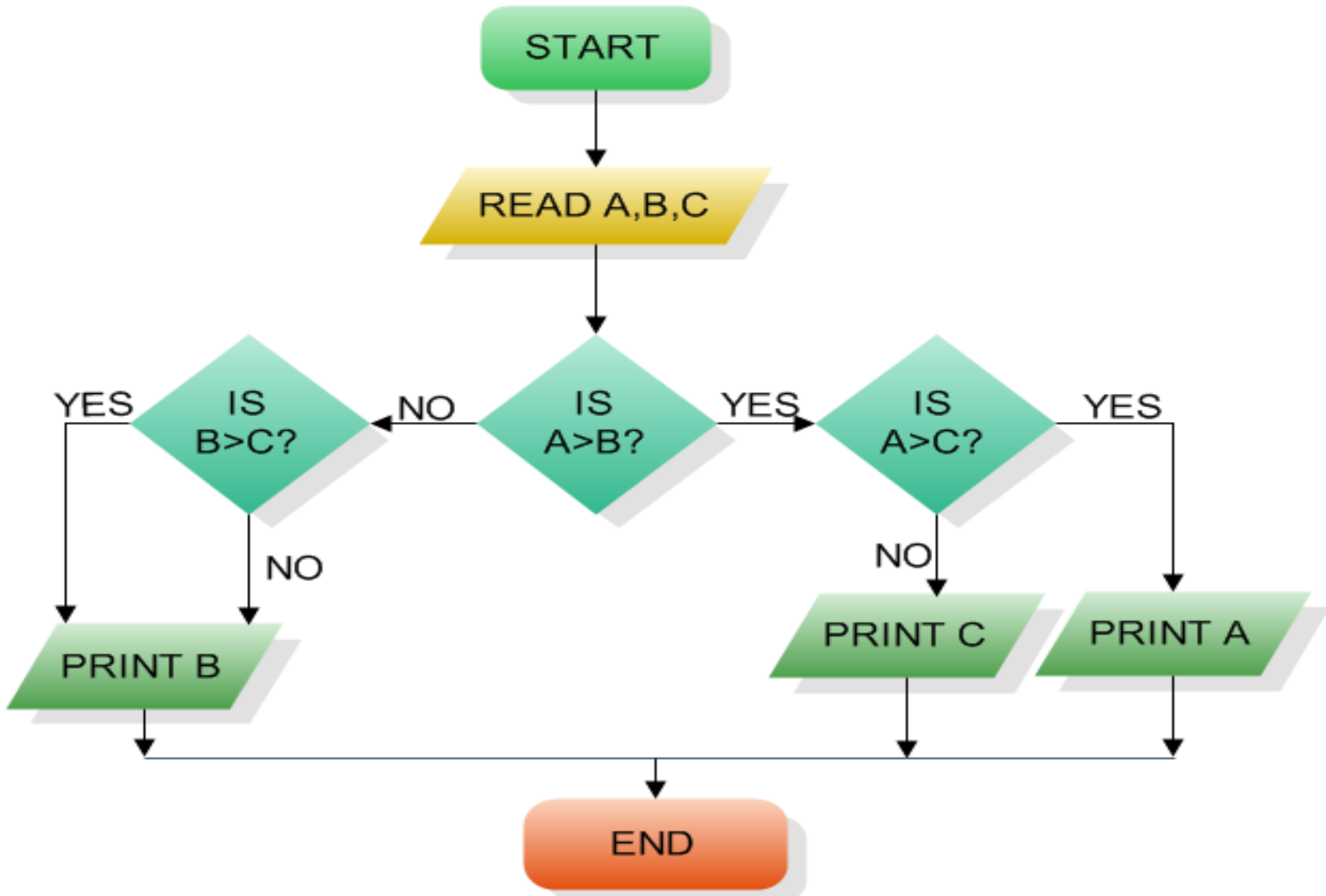
Sample Flowchart

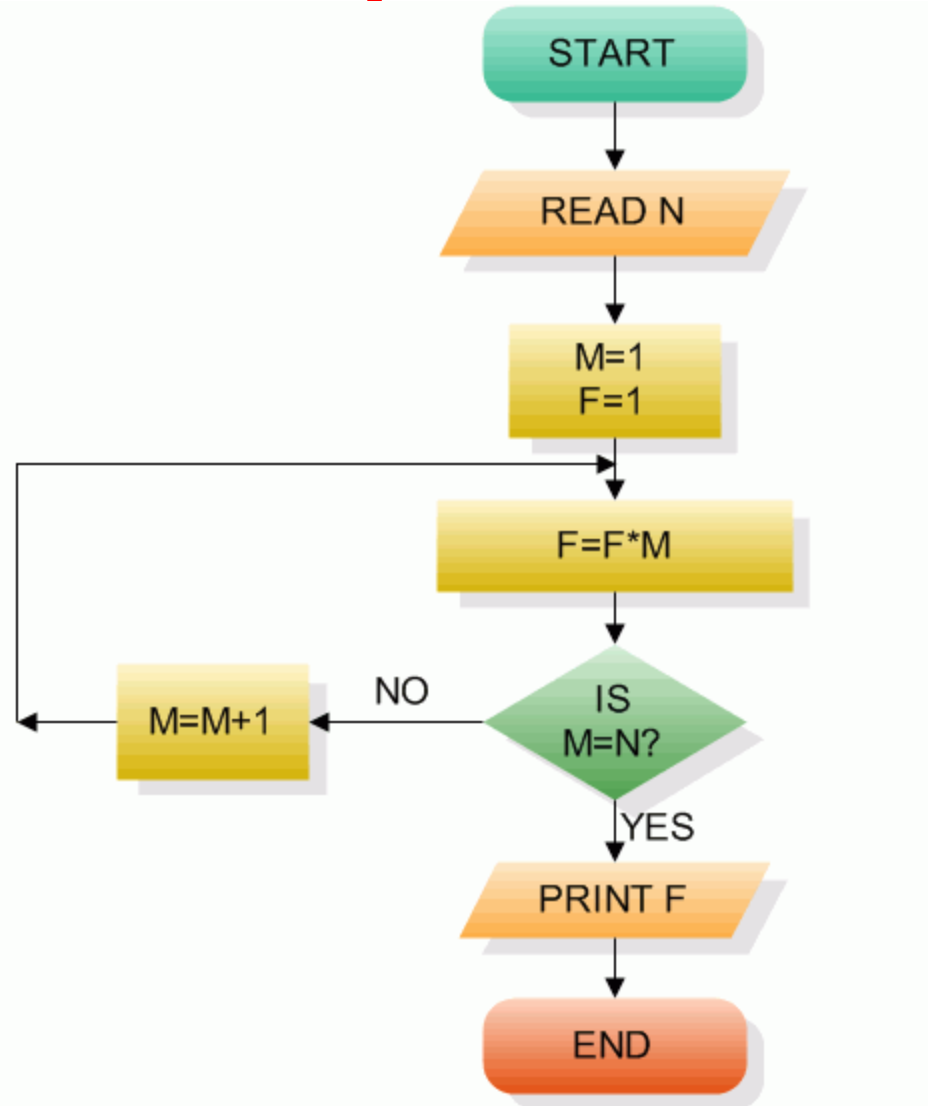- **flowchart for calculation of the roots of a quadratic equation:**

Flowchart for finding out the largest of three number
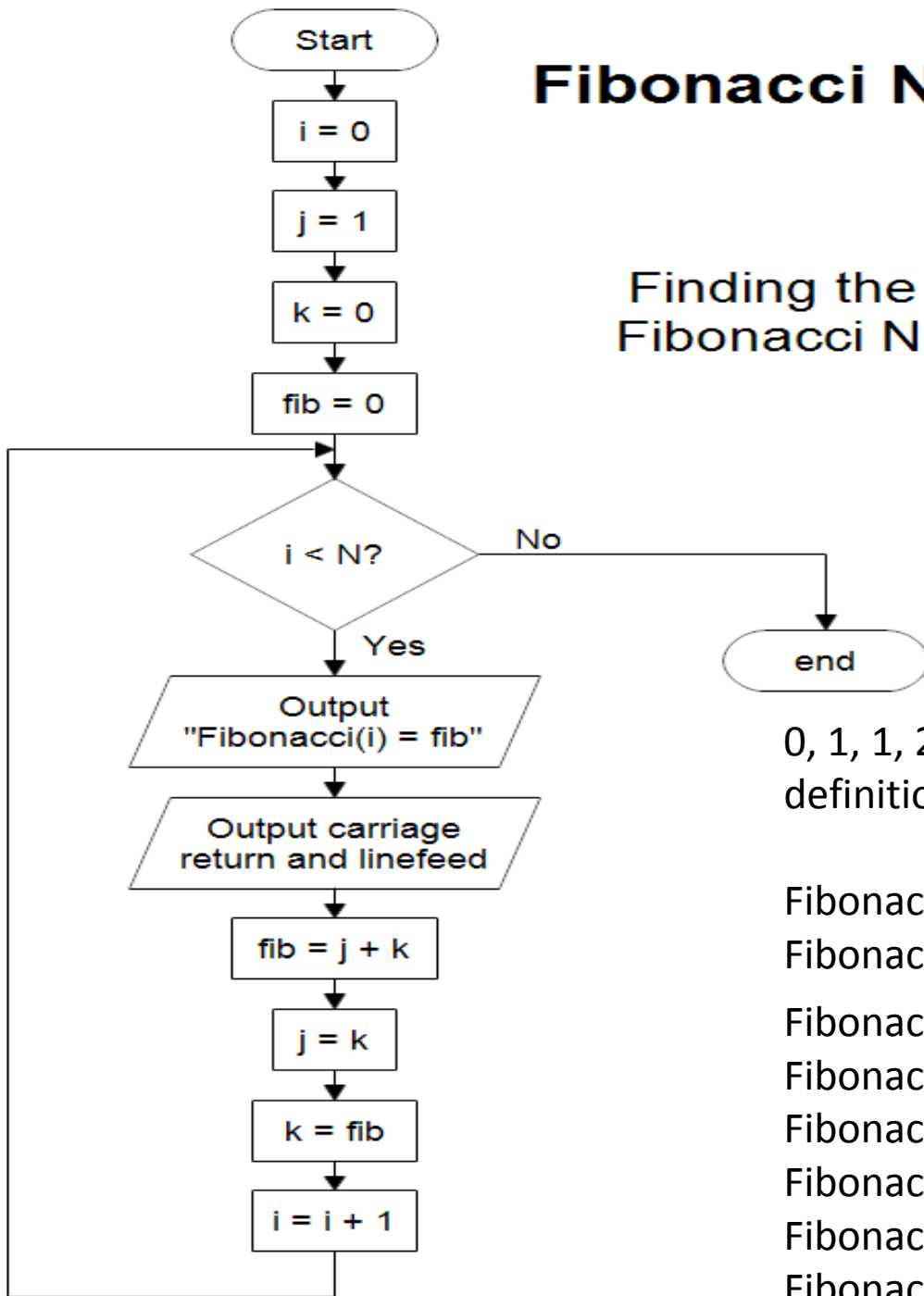
Sample Flowchart

Sample Flowchart

Flowchart for computing factorial N

Fibonacci Numbers

```
        ( Start )
           │
        ┌──────┐
        │ i = 0│
        └──────┘
           │
        ┌──────┐
        │ j = 1│
        └──────┘
           │
        ┌──────┐
        │ k = 0│
        └──────┘
           │
        ┌────────┐
        │ fib = 0│
        └────────┘
           │
    ┌──────▼──────┐
    │   i < N?    │── No ──► ( end )
    └──────┬──────┘
         Yes
           │
      ┌──────────────────────┐
     / Output                /
    / "Fibonacci(i) = fib"  /
    └──────────────────────┘
           │
      ┌──────────────────────┐
     / Output carriage       /
    / return and linefeed   /
    └──────────────────────┘
           │
        ┌──────────┐
        │ fib = j + k│
        └──────────┘
           │
        ┌──────┐
        │ j = k│
        └──────┘
           │
        ┌────────┐
        │ k = fib│
        └────────┘
           │
        ┌──────────┐
        │ i = i + 1│
        └──────────┘
```

Finding the First N Fibonacci Numbers

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ... By definition the first two numbers are:

Fibonacci(0) = 0
Fibonacci(1) = 1

Fibonacci(2) = 0 + 1 = 1
Fibonacci(3) = 1 + 1 = 2
Fibonacci(4) = 1 + 2 = 3
Fibonacci(5) = 2 + 3 = 5
Fibonacci(6) = 3 + 5 = 8
Fibonacci(7) = 5 + 8 = 13

FIGURE 118. Procedural representation of "a red cube which supports a pyramid." [Adapted from Roger Schank and Kenneth Colby, Computer Models of Thought and Language (San Francisco: W. H. Freeman, 1973), p. 172.]