CS 199 Computer Programming

Spring 2018

Lecture 2

Problem Solving

ALGORITHMS AND FLOWCHARTS

- **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

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:

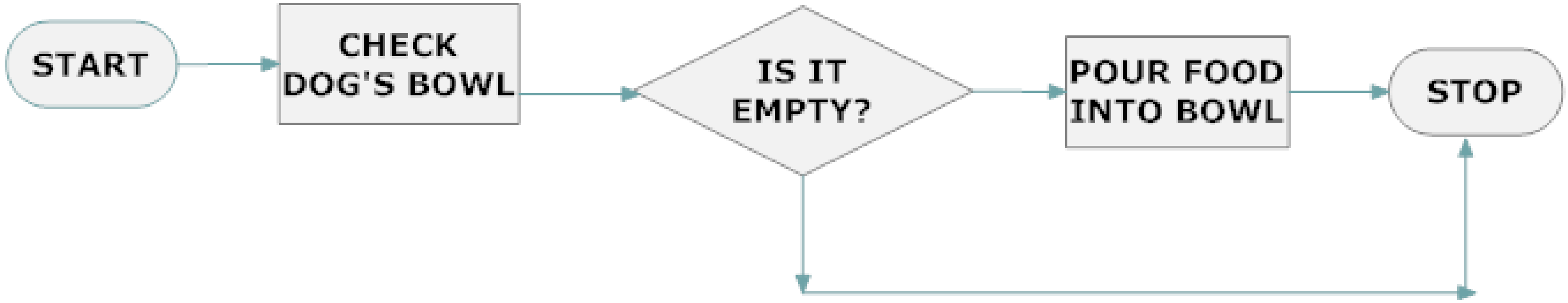
```
Input a set of 4 marks
Calculate their average by summing and
dividing by 4
if average is below 50
                Print "FAIL"
        else
                Print "PASS"
```

Detailed Algorithm

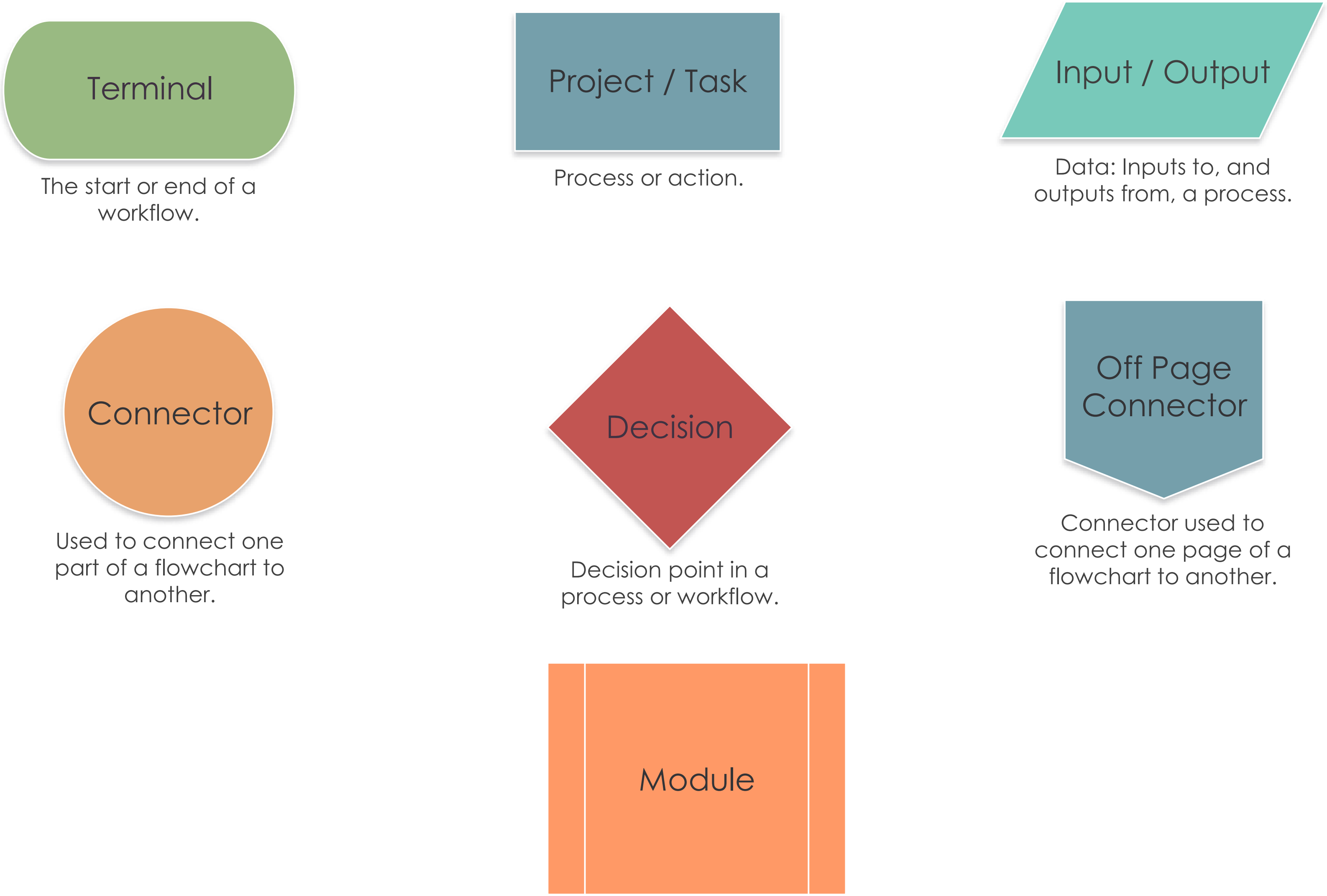
```
Step 1:  Input M1,M2,M3,M4
Step 2: GRADE ← (M1+M2+M3+M4)/4
Step 3: if (GRADE < 50) then
            Print "FAIL"
        else
            Print "PASS"
```

Flowchart

- A flowchart is a schematic representation of an algorithm or a process.
- A flowchart gives a step-by-step procedure for solution of a problem.



Basic Flowchart Shapes and Definitions



Terminal

The start or end of a workflow.

Project / Task

Process or action.

Input / Output

Data: Inputs to, and outputs from, a process.

Connector

Used to connect one part of a flowchart to another.

Decision

Decision point in a process or workflow.

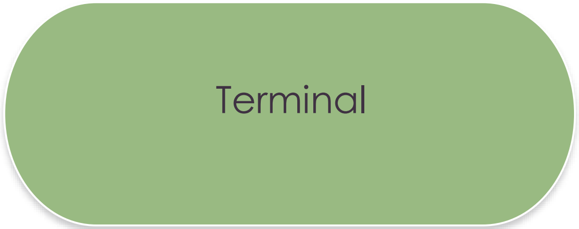
Off Page Connector

Connector used to connect one page of a flowchart to another.

Module

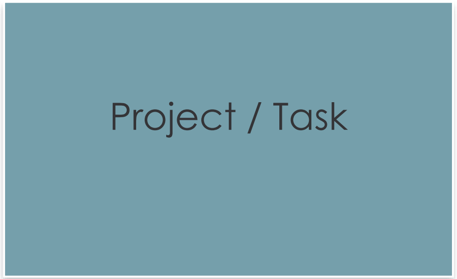
Terminal

- An oval flow chart shape indicates the start or end of the process
- Usually containing the word “Start” or “End”.



Project/Task

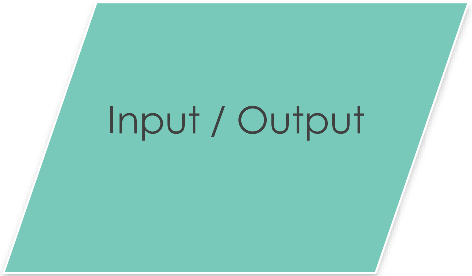
- The Process Symbol represents any process, function, or action and is the most frequently used symbol in flowcharting
- Examples include
  - Add 1
  - Turn the motor on
  - Turn the light off
  - Rotate the part



Project / Task

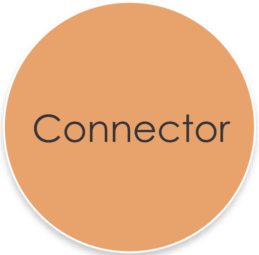
Input/Output

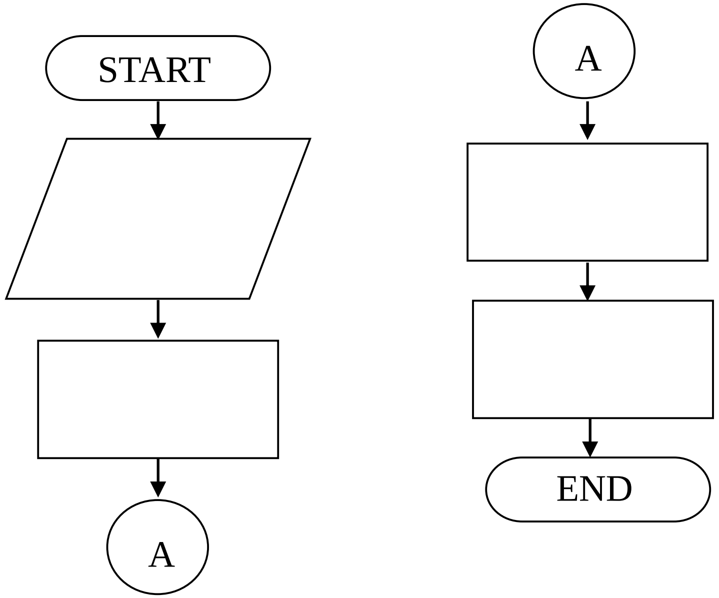
- The Input/Output Symbol represents data that is available for input or resulting from processing (i.e. customer database records)
- Examples include
  - Type in the weight
  - Check the balance
  - Time the operation



Connector

- The Connector Symbol represents the exit to, or entry from, another part of the same flow chart. It is usually used to break a flow line that will be continued elsewhere.

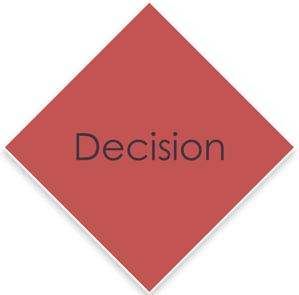




•The “A” connector indicates that the second flowchart segment begins where the first segment ends.

Decision

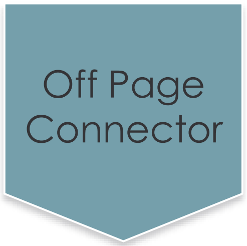
- The Decision Symbol is a junction where a decision must be made. A single entry may have any number of alternative solutions, but only one can be chosen
- Examples include
  - Is this number larger than 10?
  - Does the weight meet specifications?
  - Has the count been reached?



Decision point in a process or workflow.

Off Page Connector

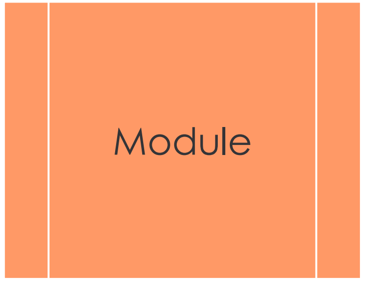
- Off-page Connector Symbols are used to indicate the flow chart continues on another page. Often the page number is placed in the shape for easy reference

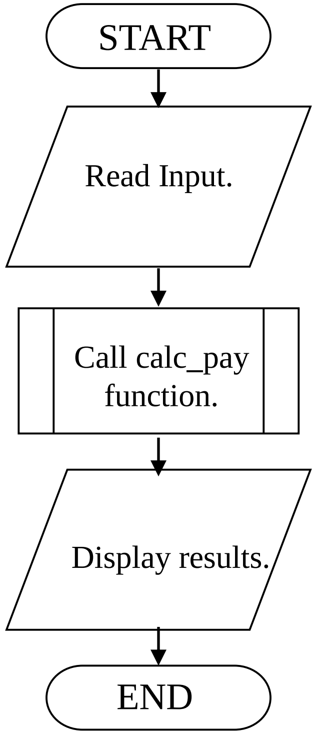


Connector used to connect one page of a flowchart to another.

Module

- The position of the module symbol indicates the point the module is executed.
- A separate flowchart can be constructed for the module.



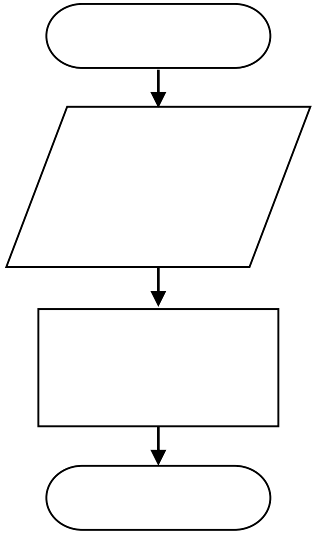


Three Flowchart Structures

- Sequence
- Decision
- Repetition

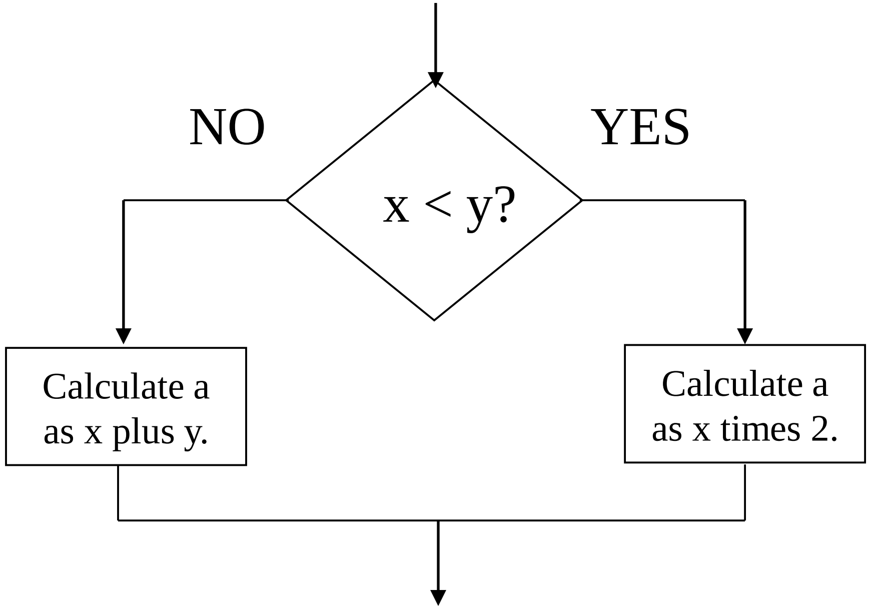
Sequence Structure

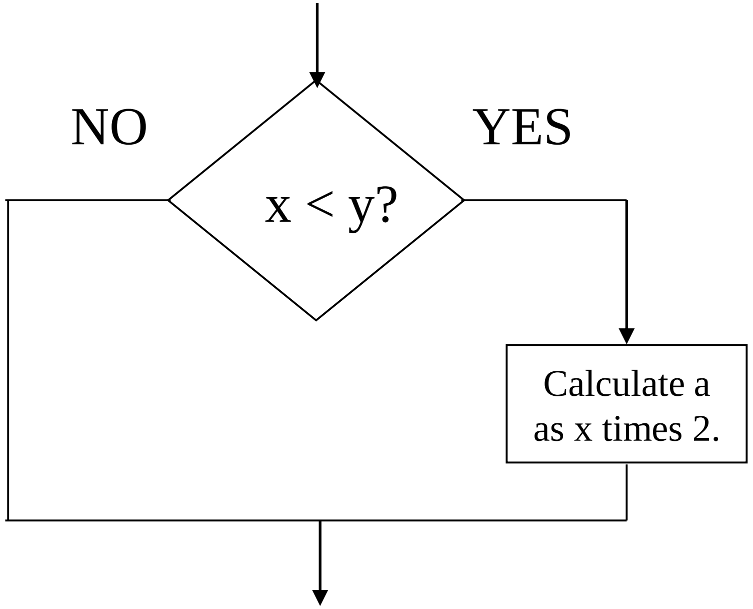
- A series of actions are performed in sequence



Decision Structure

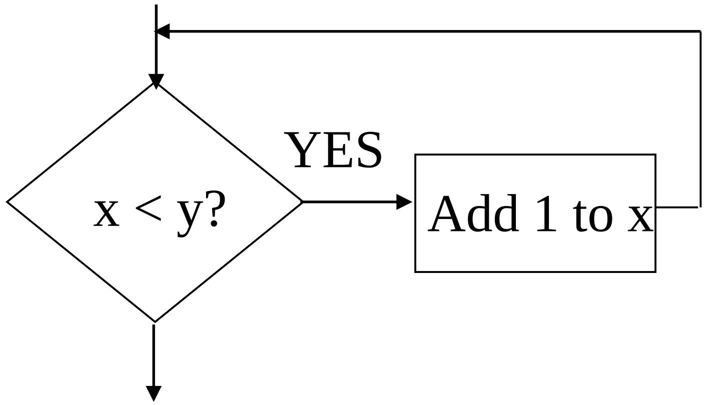
- A sequence of actions executed based on condition.





Repetition Structure

- A sequence of actions executed many times.



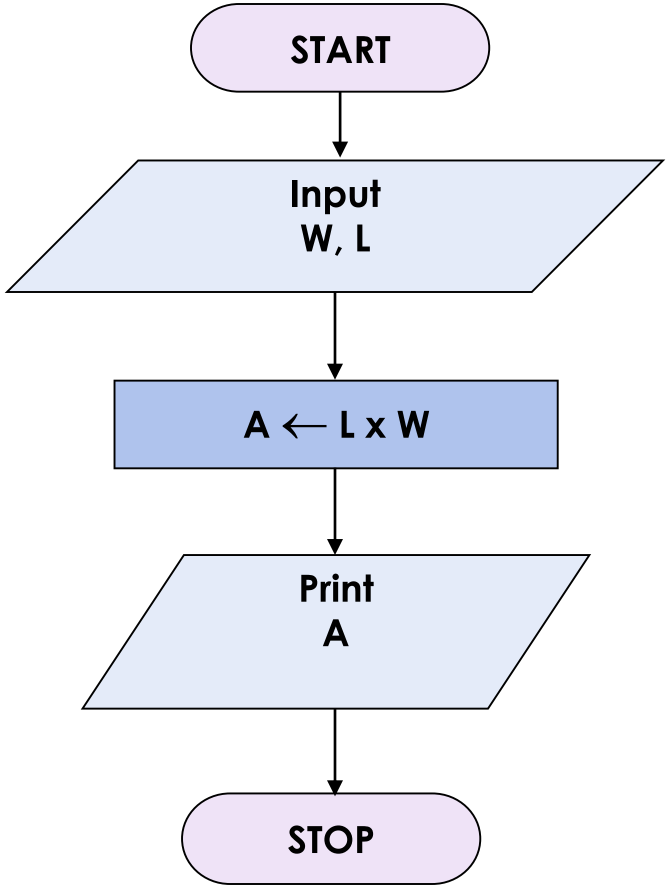
1 is added to variable x until x > y

Example 1

- 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 1(cont.)

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



Example 2

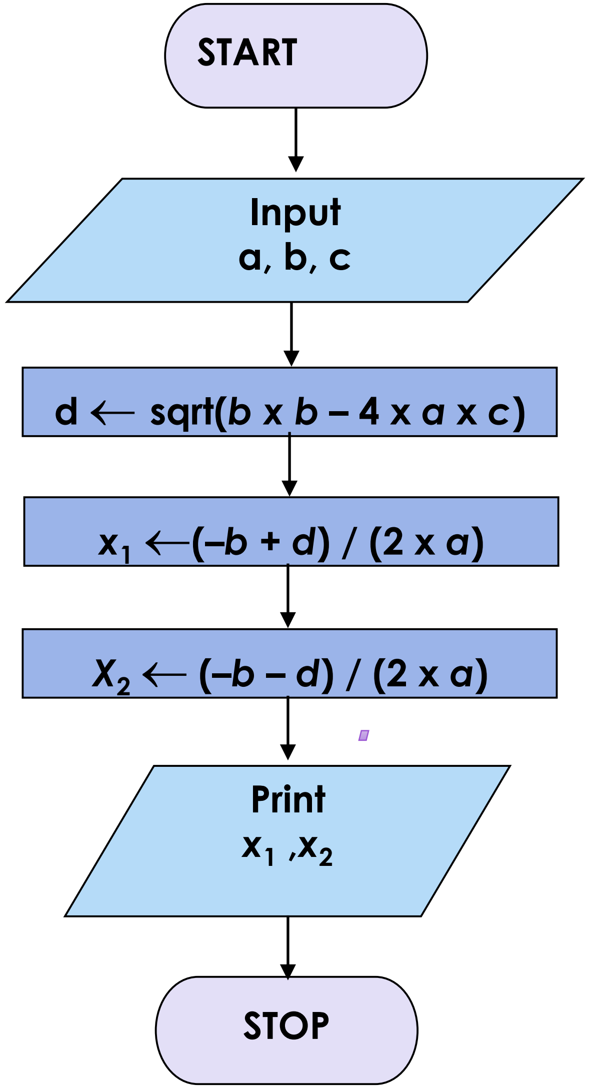
- Write an algorithm and draw a flowchart that will calculate the roots of a quadratic equation

$$ax^2 + bx + c = 0$$

- Hint: d = sqrt ($b^2 - 4ac$), and the roots are: x1 = (–b + d)/2a and x2 = (–b – d)/2a

Example 2 (cont.)

- Algorithm:
  - Step 1: Input a, b, c
  - Step 2: d ← sqrt ($b \times b - 4 \times a \times c$)
  - Step 3: x1 ← (–b + d) / (2 x a)
  - Step 4: x2 ← (–b – d) / (2 x a)
  - Step 5: Print x1, x2



START

Input
a, b, c

d ← sqrt(*b* x *b* – 4 x *a* x c)

x_1 ←(–*b* + *d*) / (2 x *a*)

X_2 ← (–*b* – *d*) / (2 x *a*)

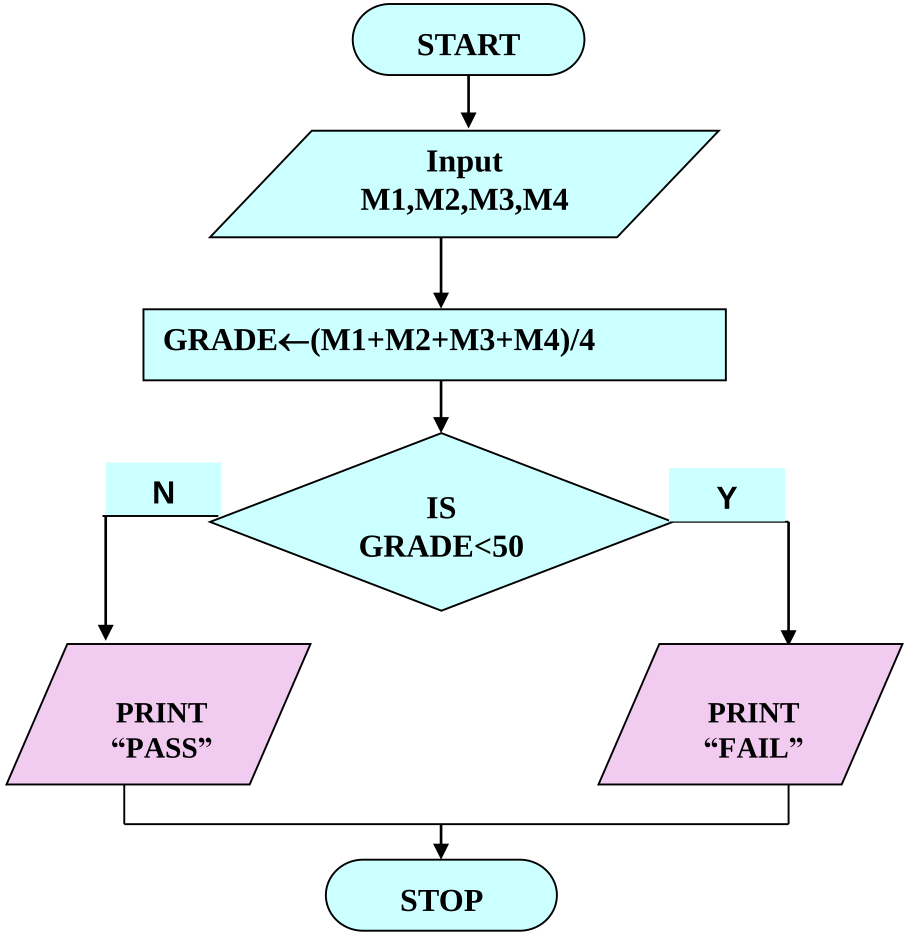
Print
x_1 ,x_2

STOP

Example 3

- 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.

Example 3 (cont.)



- 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
```

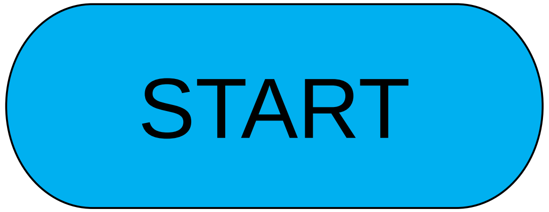
Example 4

- Count from 1 to 100 by odd numbers.
- Before attempting to draw the flowchart, determine what you want the output to be.
- What is the first block (always)?

COUNT...

Step 1

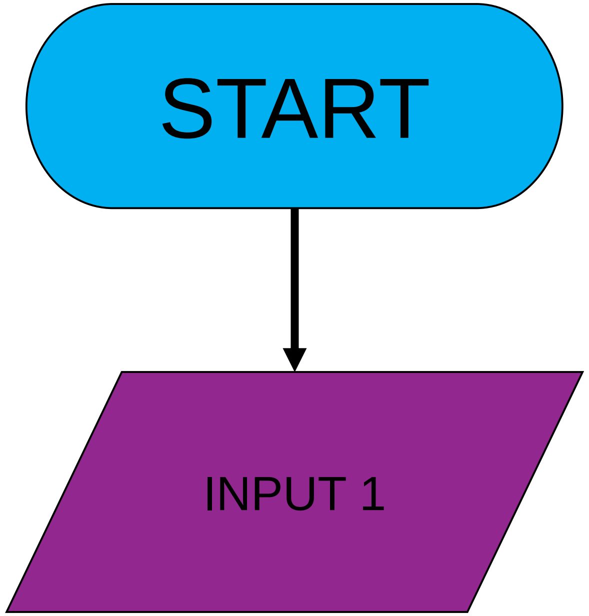
- The output will be 1, 3, 5, 7, 9 ……99 .
- The Start block is always first.



START

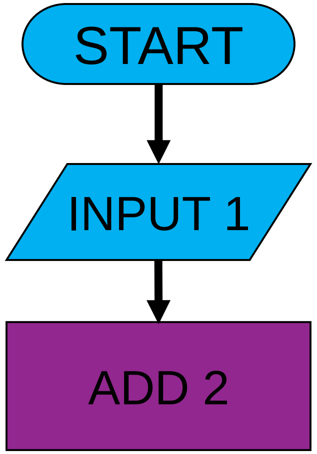
Step 2

The program begins with the number 1.



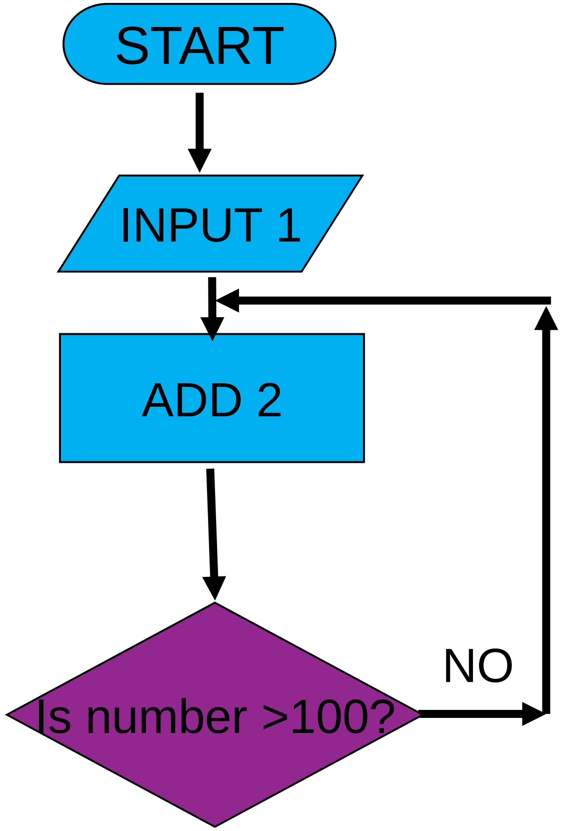
Step 3

The number 2 will be added to 1 so that the program will continue to count by odd numbers.



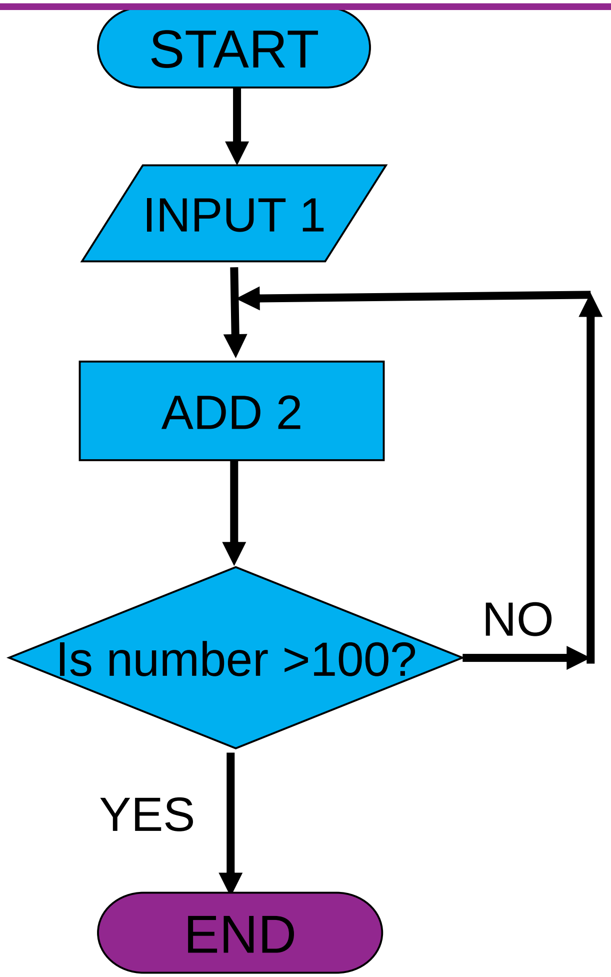
Step 4

Add a decision block so that the program will continue counting until the value is greater than 100.



Step 5

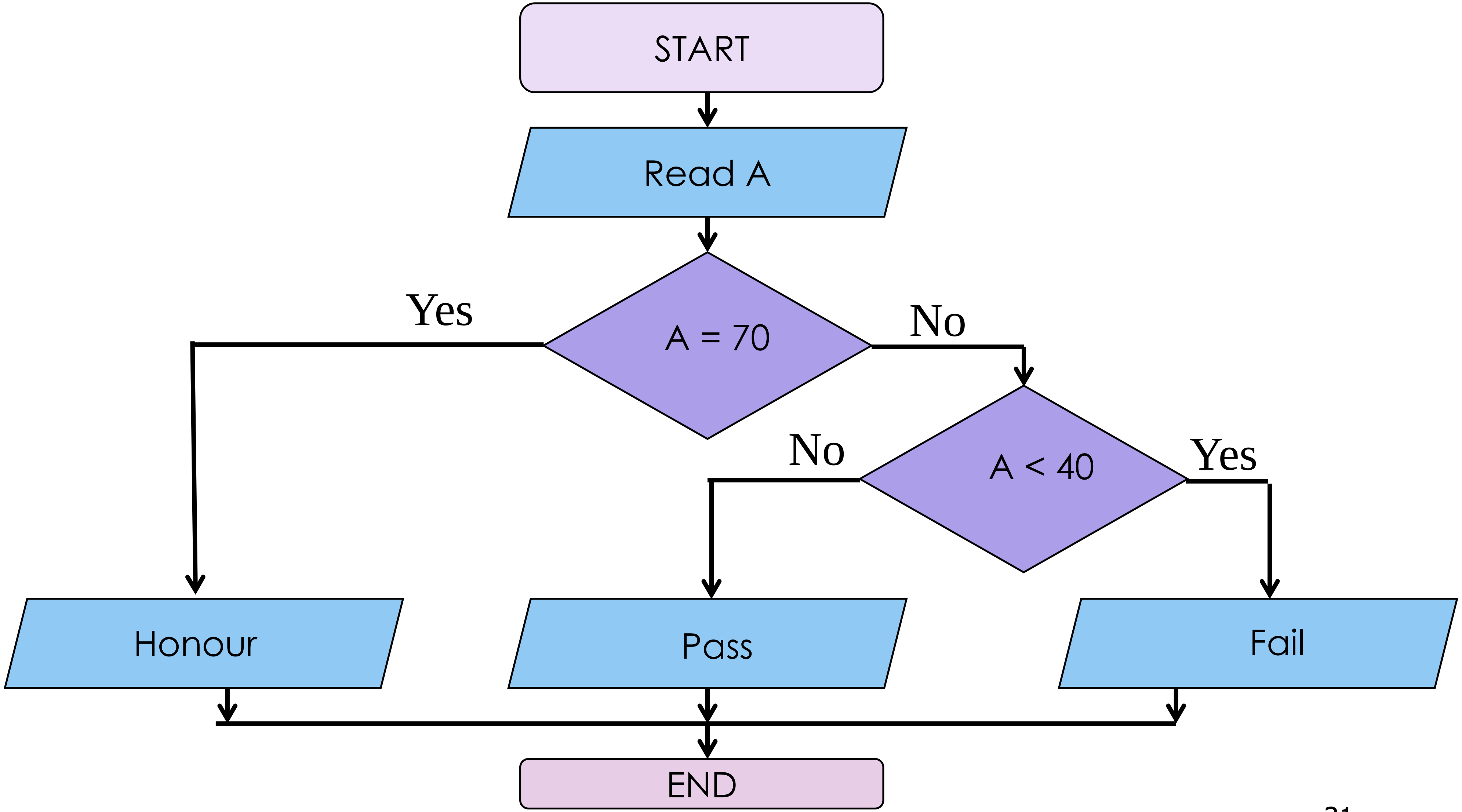
Once the number is greater than 100, the program ends.



Example 4

- We want to create a flowchart that prints out the word “Honour” if the number input is 70, if the number is less than 40 print out the word “Fail”, otherwise print out the word “Pass”.

Example 4 (cont.)



Example 5

- Express an algorithm to get two numbers from the user (dividend and divisor), testing to make sure that the divisor number is not zero, and displaying their quotient using a flowchart.

Example 5 (cont.)

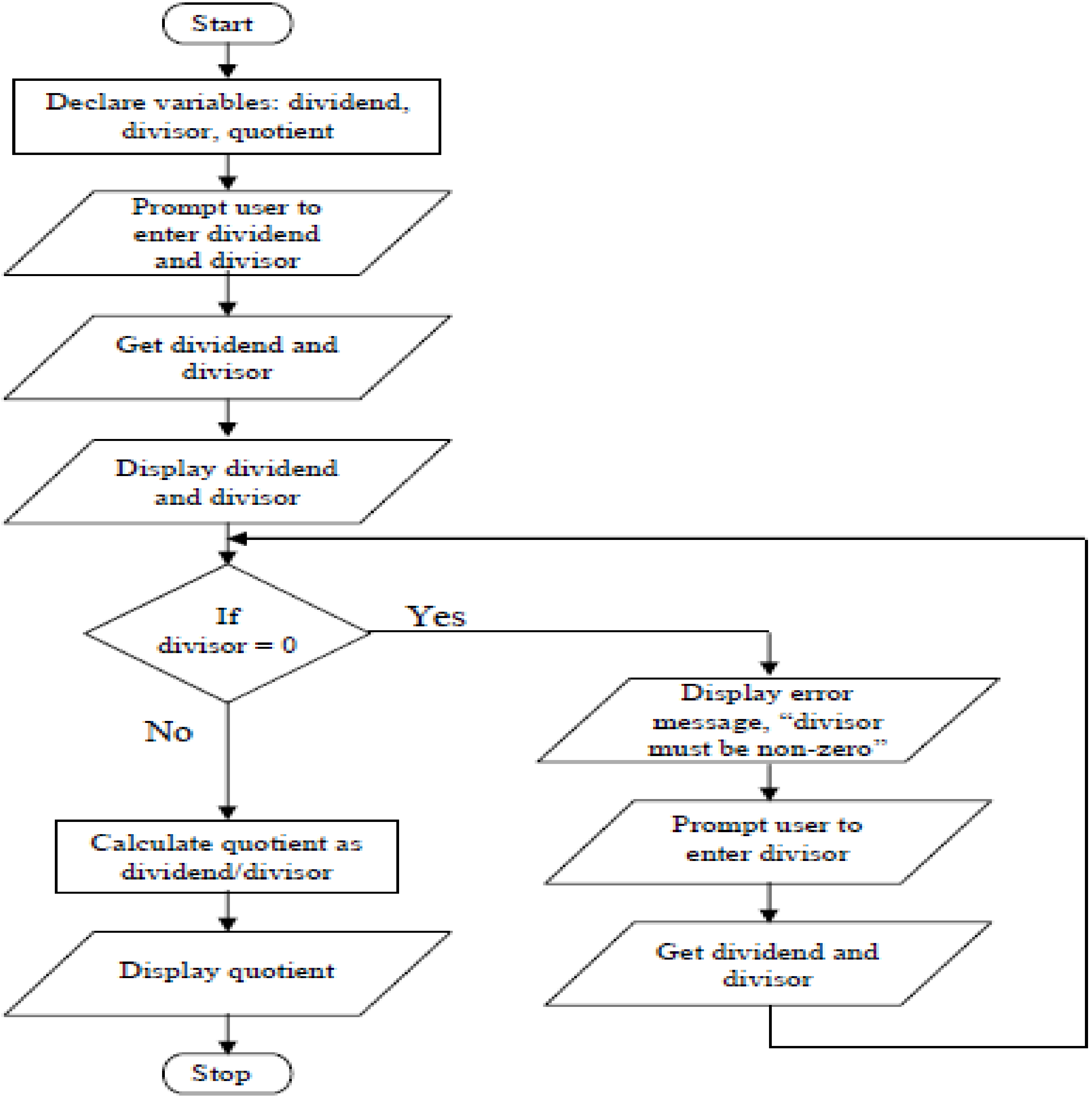
- Step 1 – Declare variables – dividend, divisor, quotient
- Step 2 – Prompt user to get dividend
- Step 3 – Store values in dividend variable
- Step 4 – Prompt user to get divisor
- Step 5 – Store value in divisor variable
- Step 6 – Display dividend and divisor
- Step 7 - Loop

  Selection: If divisor is equal to zero

  Display error message, "divisor must be non-zero" and

  go back to step 4

- Step 8 - Calculate quotient as dividend/divisor
- Step 9 - Display quotient


Start
Declare variables: dividend, divisor, quotient
Prompt user to enter dividend and divisor
Get dividend and divisor
Display dividend and divisor
If divisor = 0
Yes
No
Display error message, "divisor must be non-zero"
Prompt user to enter divisor
Get dividend and divisor
Calculate quotient as dividend/divisor
Display quotient
Stop

Example 6

- Write and algorithm and draw a flowchart to
  a) read an employee name (NAME), overtime hours worked (OVERTIME), hours absent (ABSENT) and
  b) determine the bonus payment (PAYMENT).

Bonus Schedule	
OVERTIME – (2/3)*ABSENT	Bonus Paid
>40 hours	$50
>30 but ≤ 40 hours	$40
>20 but ≤ 30 hours	$30
>10 but ≤ 20 hours	$20
≤ 10 hours	$10

Feedback request

- **Please mail questions and constructive comments to**

  marwa.elmenyawi@bhit.bu.edu.eg

- **Your feedback will be most appreciated**
  - On style, contents, detail, examples, clarity, conceptual problems, exercises, missing information, depth, etc.

The next lecture

- **Will talk about how to write your first program in Fortran.**

Thank You