Problem Solving and Programming

CSE1001

Prof. Tulasi Prasad Sariki

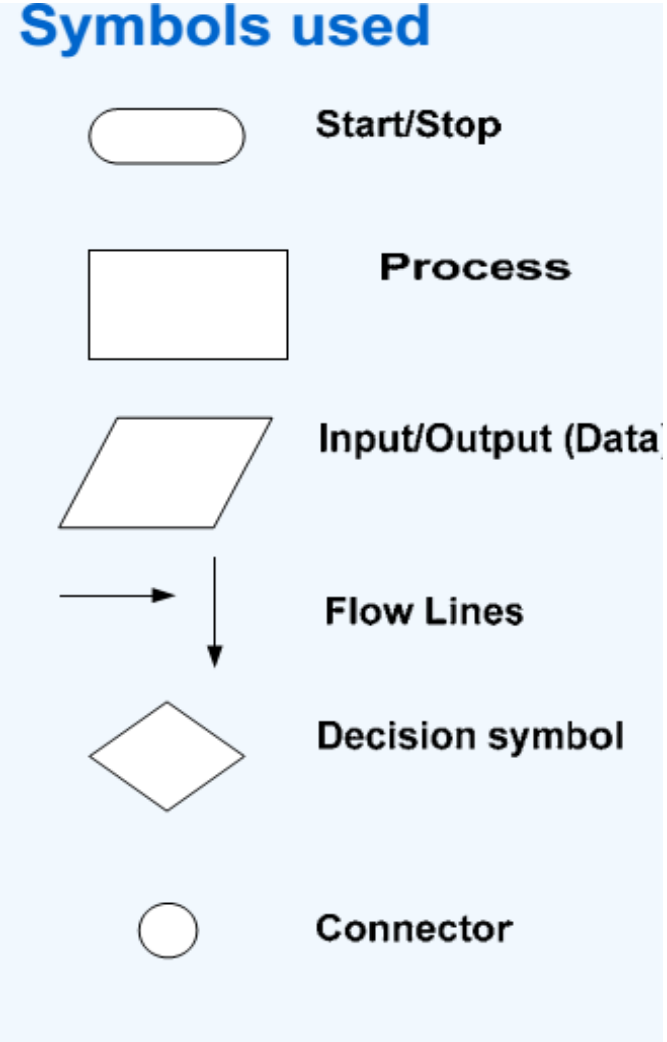
Flow Chart

Drawing Flowcharts

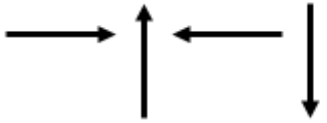
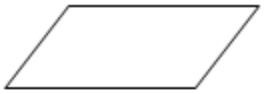
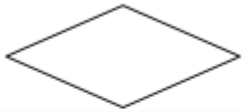
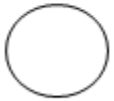
- Flowchart is the graphic representations of the individual steps or actions to implement a particular module.
- Flowchart can be linked to the blueprint of a building.
- An architect draws a blueprint before construction of a building, so the programmer draws a flowchart before writing a program.
- Flowchart is independent of any programming language.

Flowcharts

A flow chart is an organized combination of shapes, lines and text that graphically illustrate a process or structure.



Pre-Programming Phase

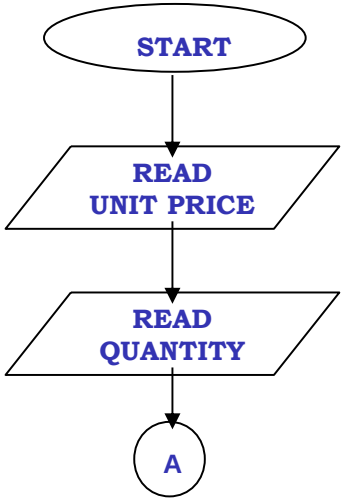
Symbol	Function
	Show the direction of data flow or logical solution.
	Indicate the beginning and ending of a set of actions or instructions (logical flow) of a module or program.
	Indicate a process, such as calculations, opening and closing files.
	Indicate input to the program and output from the program.
	Use for making decision. Either True or False based on certain condition.
	Use for doing a repetition or looping of certain steps.
	Connection of flowchart on the same page.
	Connection of flowchart from page to page.

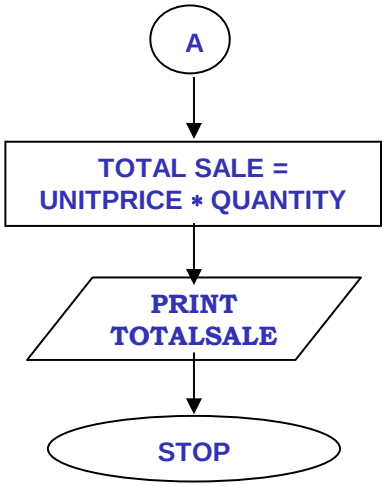
Sequential Logical Structure

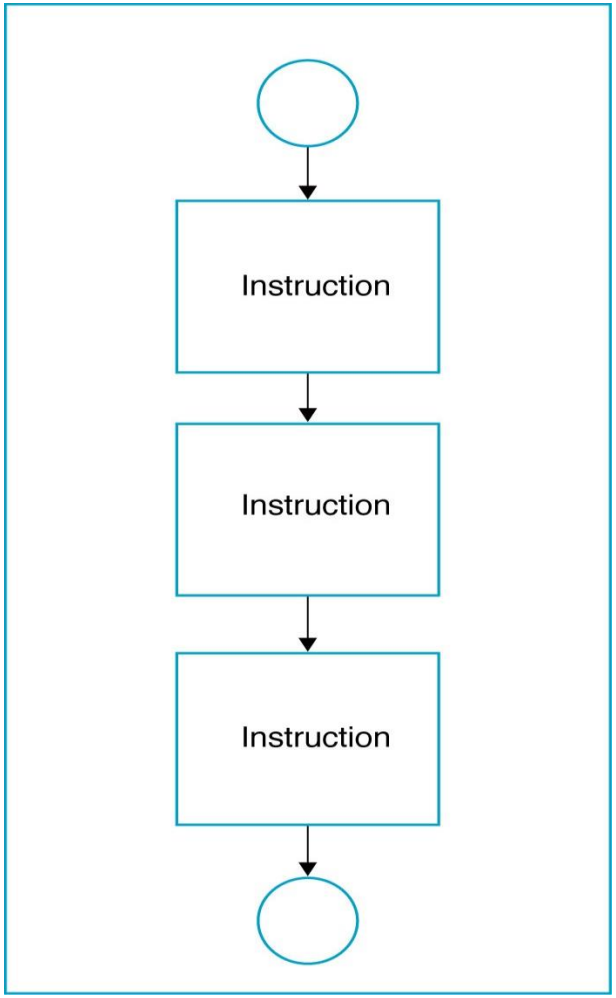
Given the unit price of a product and the quantity of the product sold, draw a flowchart to calculate and print the total sale.

Solution: Stepwise Analysis of the Sale Problem

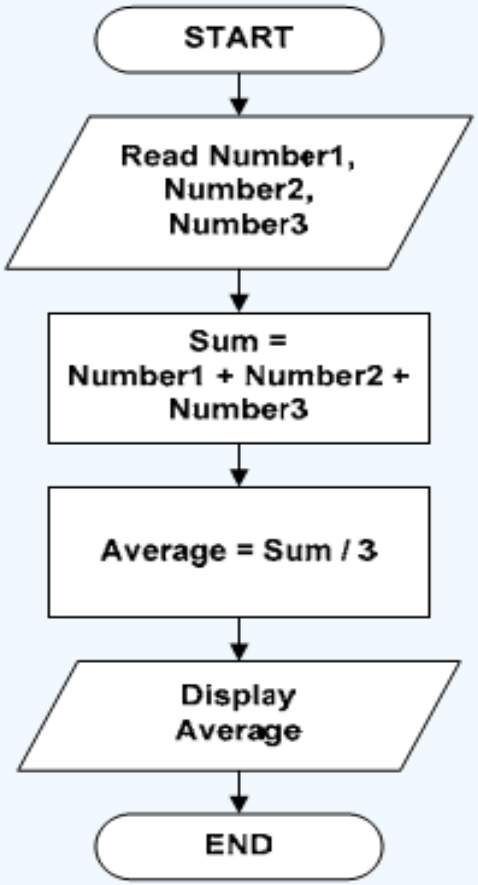
- Read the unit price and the quantity
- Calculate total sale = unit price and quantity
- Print total sale



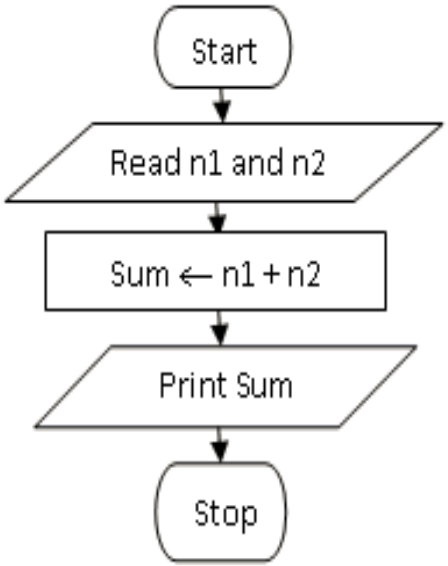




Flowchart Examples



Average



Sum of 2 Numbers

Flowchart for Conditional Expressions

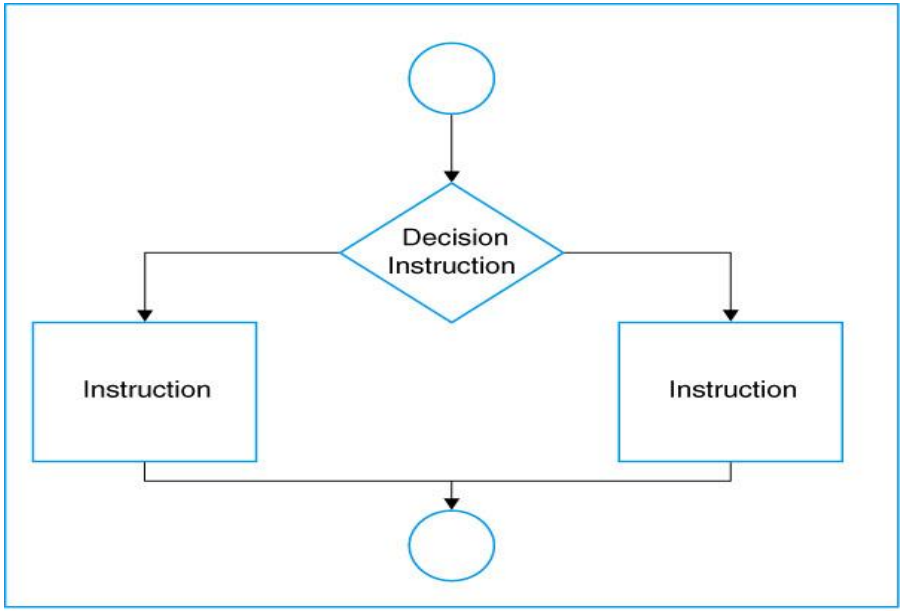
- Implements using the IF/THEN/ELSE instruction.
- Tells the computer that IF a condition is true, THEN execute a set of instructions, or ELSE execute another set of instructions.
- ELSE part is optional, as there is not always a set of instructions if the conditions are false.
- Algorithm:

IF <condition(s)> THEN

<TRUE instruction(s)>

ELSE

<FALSE instruction(s)

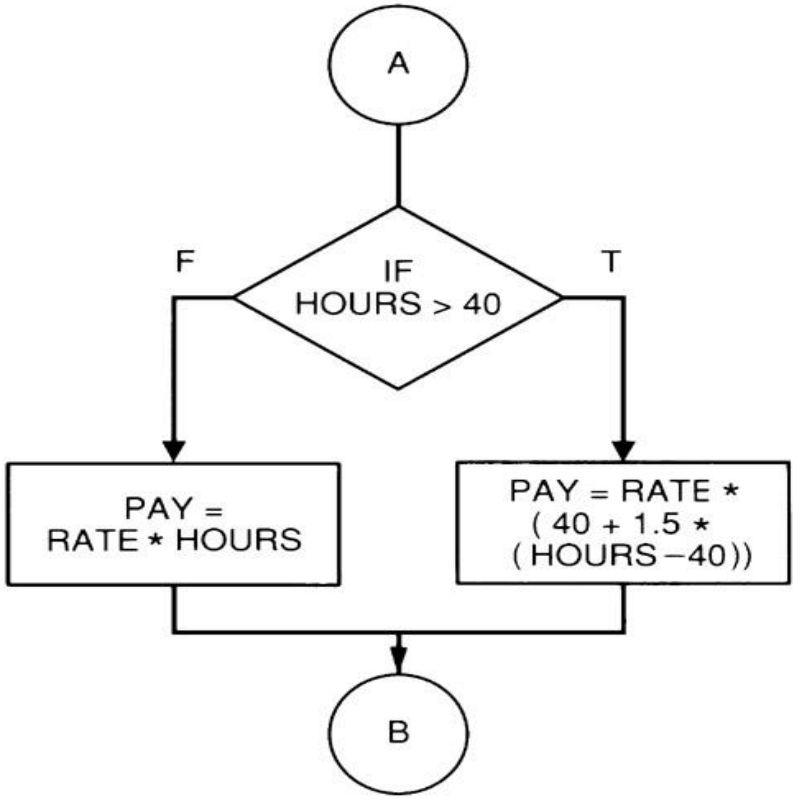


Examples of Conditional Expressions

- A < B (A and B are the same data type – either numeric, character, or string)
- X + 5 >= Z (X and Z are numeric data)
- E < 5 or F > 10 (E and F are numeric data)
- DATAOK (DATAOK – logical datum)

- Assume your are calculating pay at an hourly rate, and overtime pay(over 40 hours) at 1.5 times the hourly rate.
  - IF the hours are greater than 40, THEN the pay is calculated for overtime, or ELSE the pay is calculated in the usual way.

Flowchart for Pay Calculations

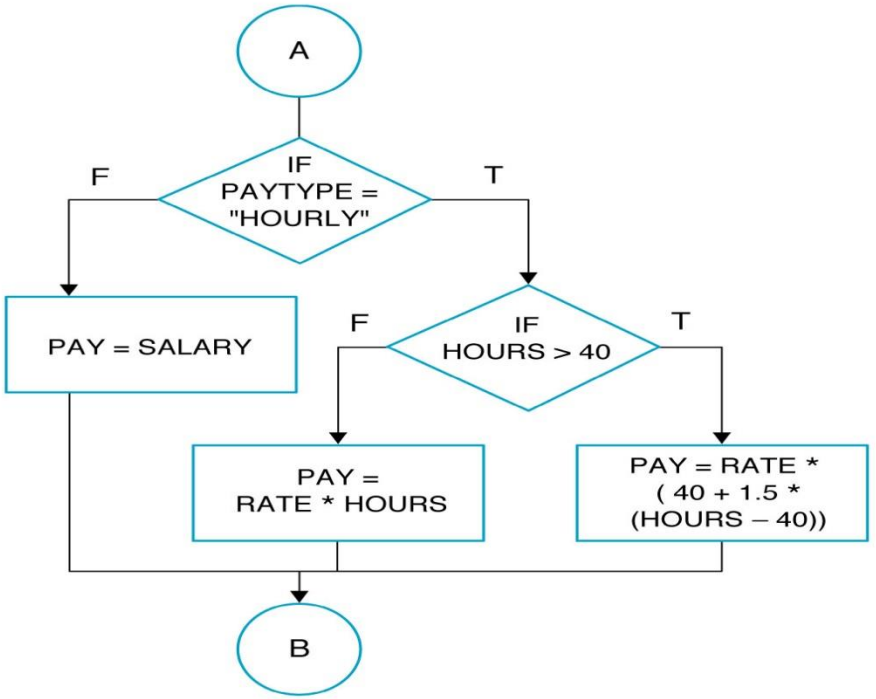
Algorithm	Flowchart
IF HOURS > 40 T THEN PAY = RATE * (40 + 1.5 * (HOURS − 40)) F ELSE PAY = RATE * HOURS	

Note: For all flowcharts with decision blocks, T = TRUE and F = FALSE

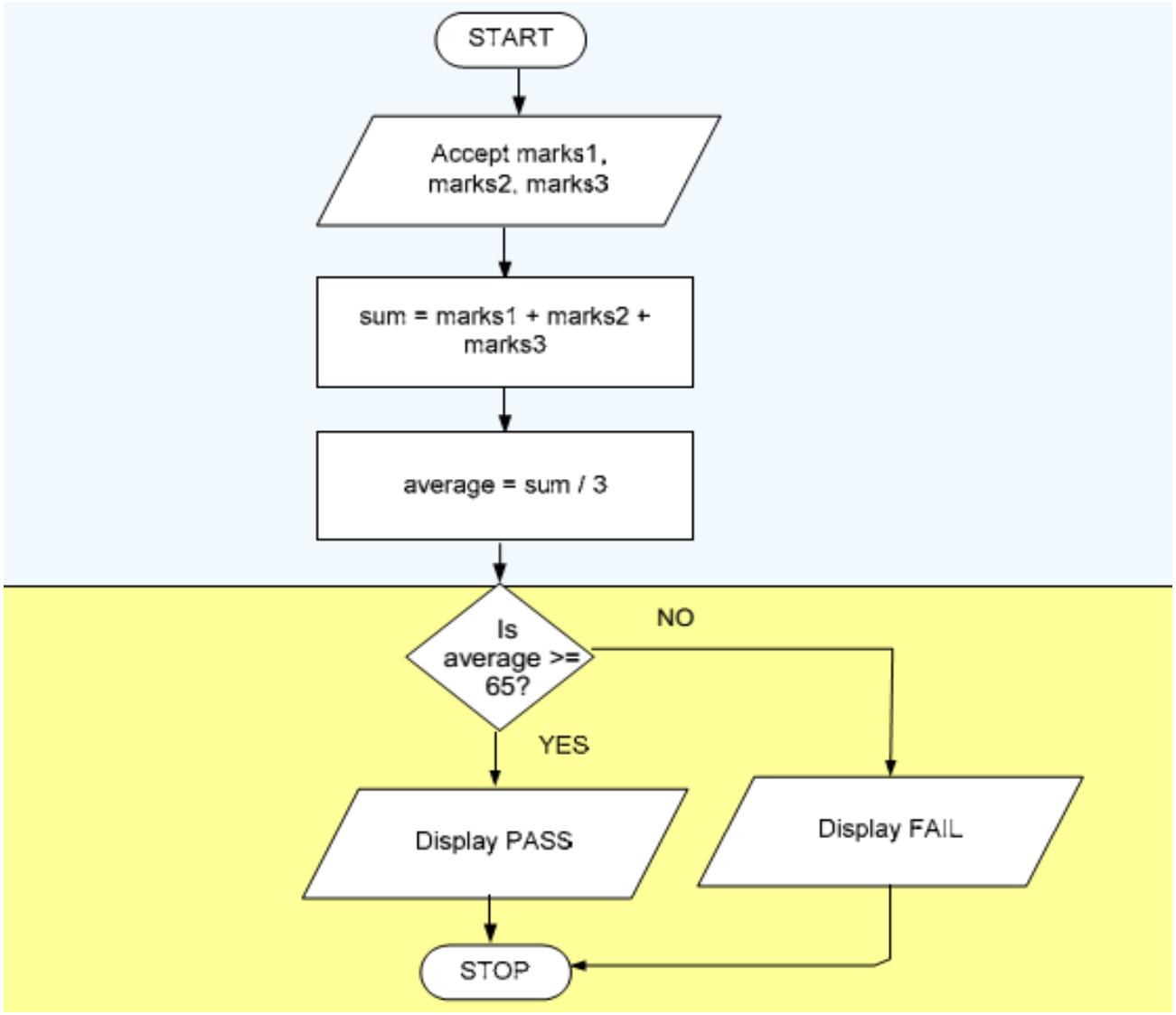
Flowchart for Pay Calculations

Nested IF/THEN/ELSE INSTRUCTIONS

- Multiple decisions.
- Instructions are sets of instruction in which each level of a decision is embedded in a level before it.

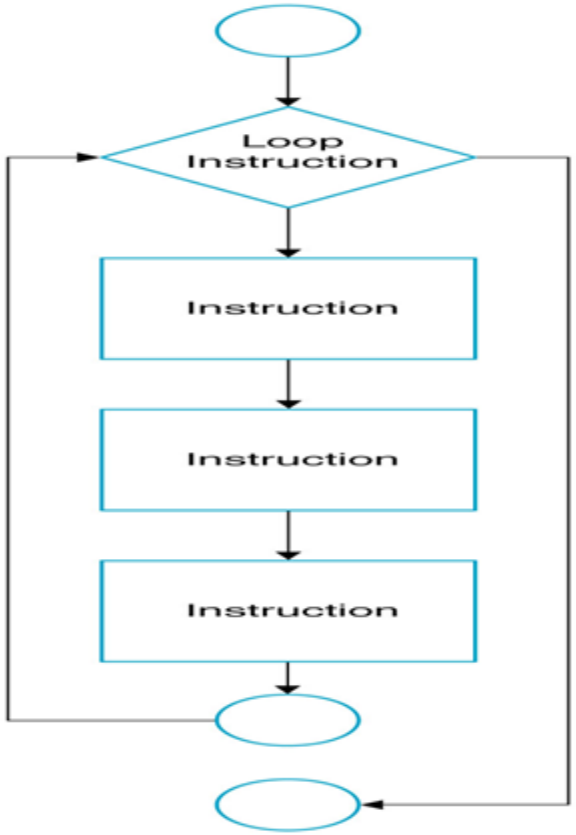
Algorithm	Flowchart
IF PAYTYPE = "HOURLY" THEN T IF HOURS > 40 T THEN PAY = RATE * (40 + 1.5 * (HOURS – 40)) F ELSE PAY = RATE * HOURS F ELSE PAY = SALARY	

Flowchart for Selections



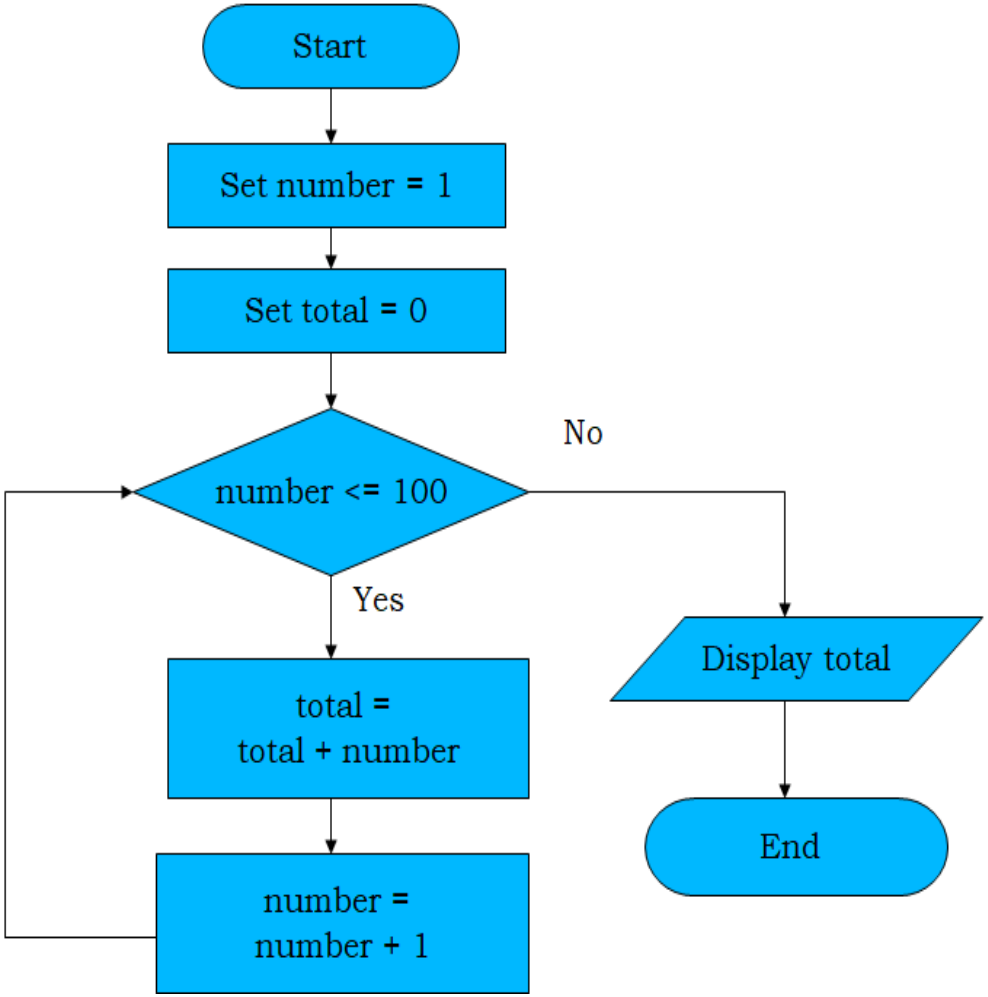
Flowchart for Iterational Structure

- Repeat structure
- To solve the problem that doing the same task over and over for different sets of data.
- Types of loop:
  - WHILE loop
  - Do..WHILE loop
  - Automatic-Counter Loop



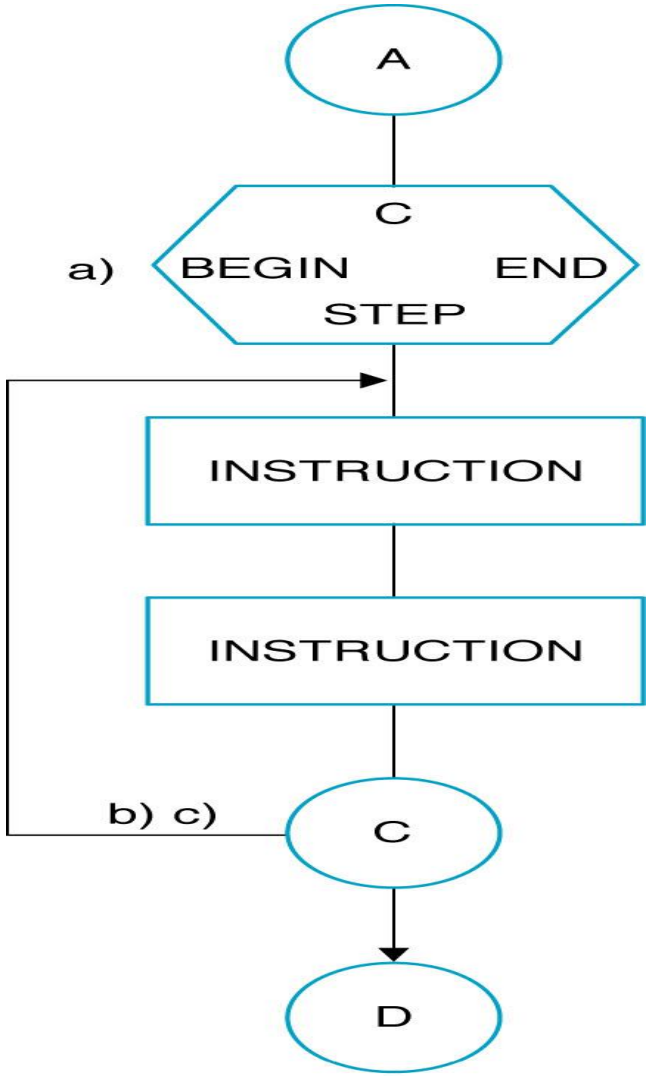
Flowchart for While Loop

- Do the loop body if the condition is ***true***.
- Example: Get the sum of 1, 2, 3, …, 100.
  - ***Algorithm:***
    - Set the number = 1
    - Set the total = 0
    - While (number <= 100)
      - total = total + number
      - number = number + 1
    - End While
    - Display total



Flowchart for Automatic Loop

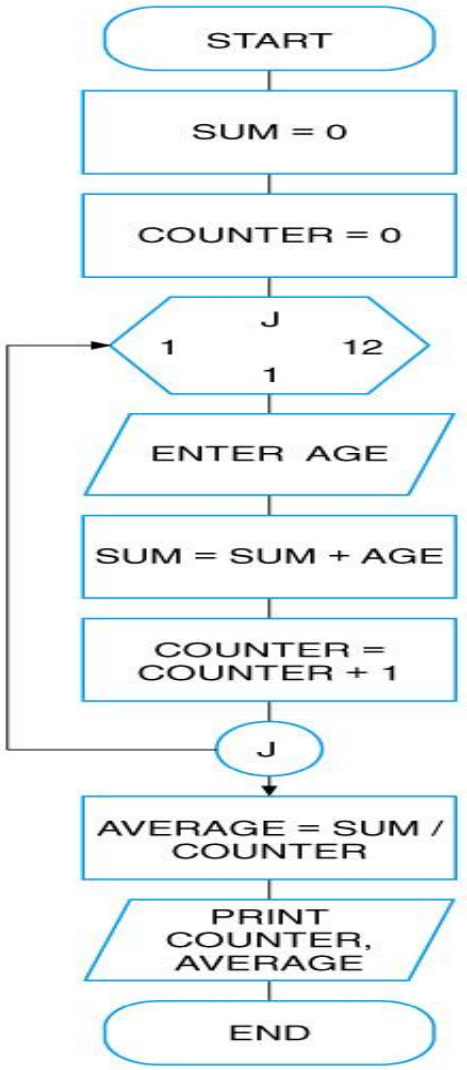
- Use variable as a counter that starts counting at a specified number and increments the variable each time the loop is processed.
- The beginning value, the ending value and the increment value may be constant.
- They should not be changed during the processing of the instruction in the loop.



Flowchart for Automatic Loop

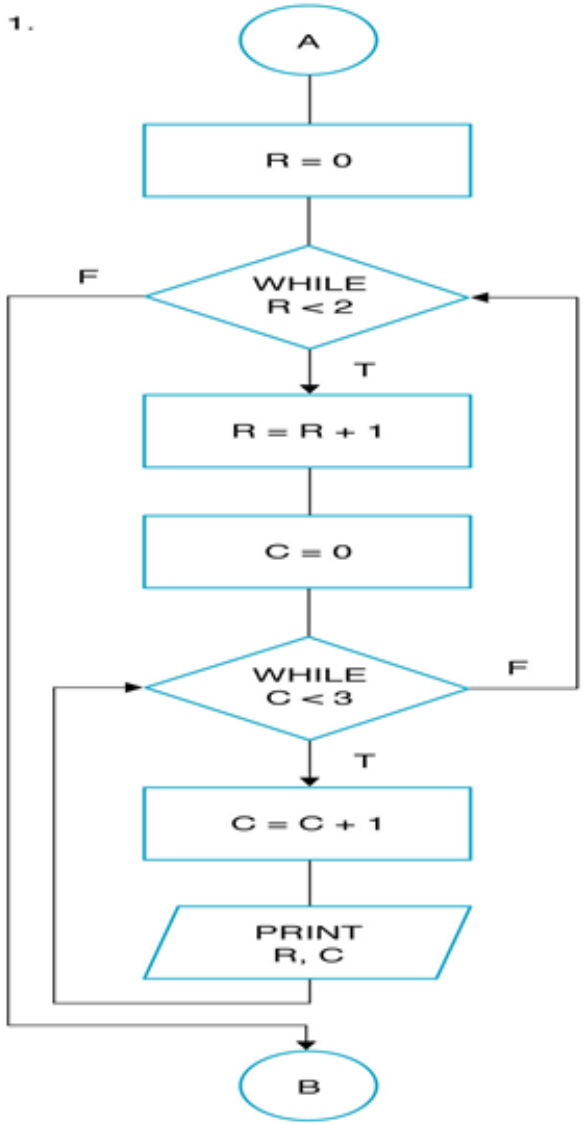
Algorithm	Flowchart

```
AVERAGE AGE
1. SUM = 0
2. COUNTER = 0
3. LOOP: J = 1 TO 12
       SUM = SUM + AGE
       COUNTER = COUNTER + 1
   LOOP-END: J
4. AVERAGE = SUM/COUNTER
5. PRINT COUNTER, AVERAGE
6. END
```

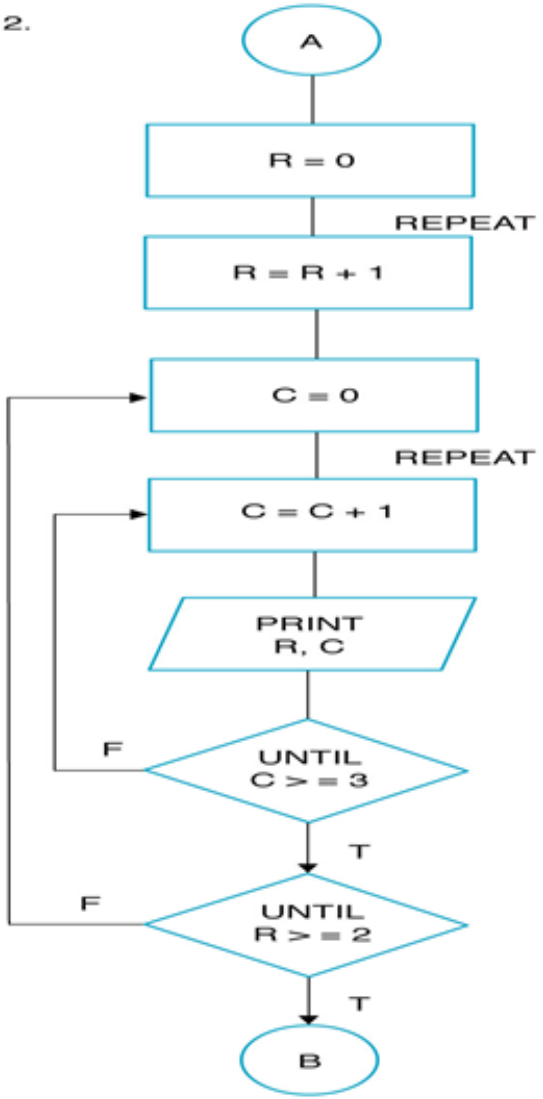


Flowchart for Nested Loop

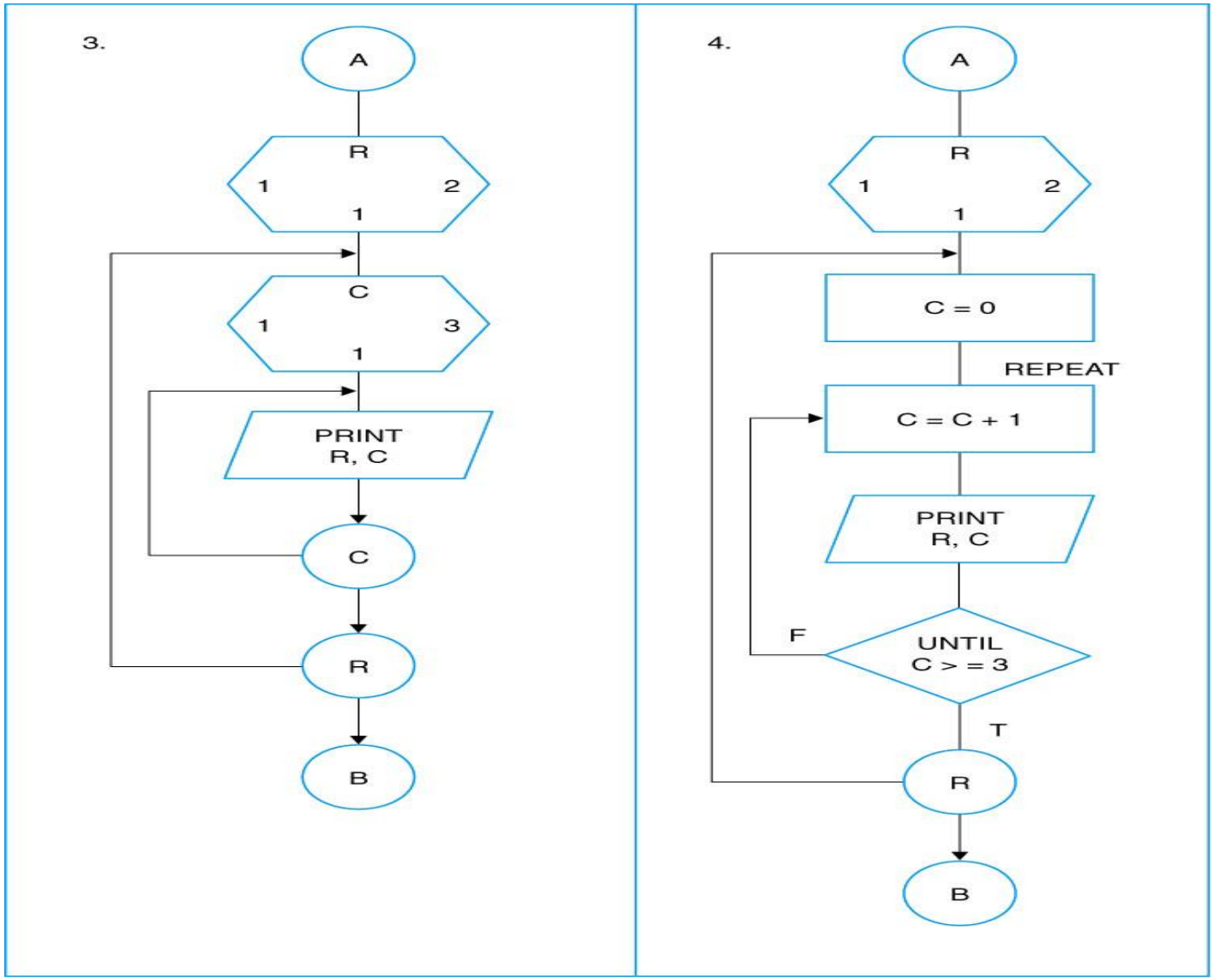
1.



2.

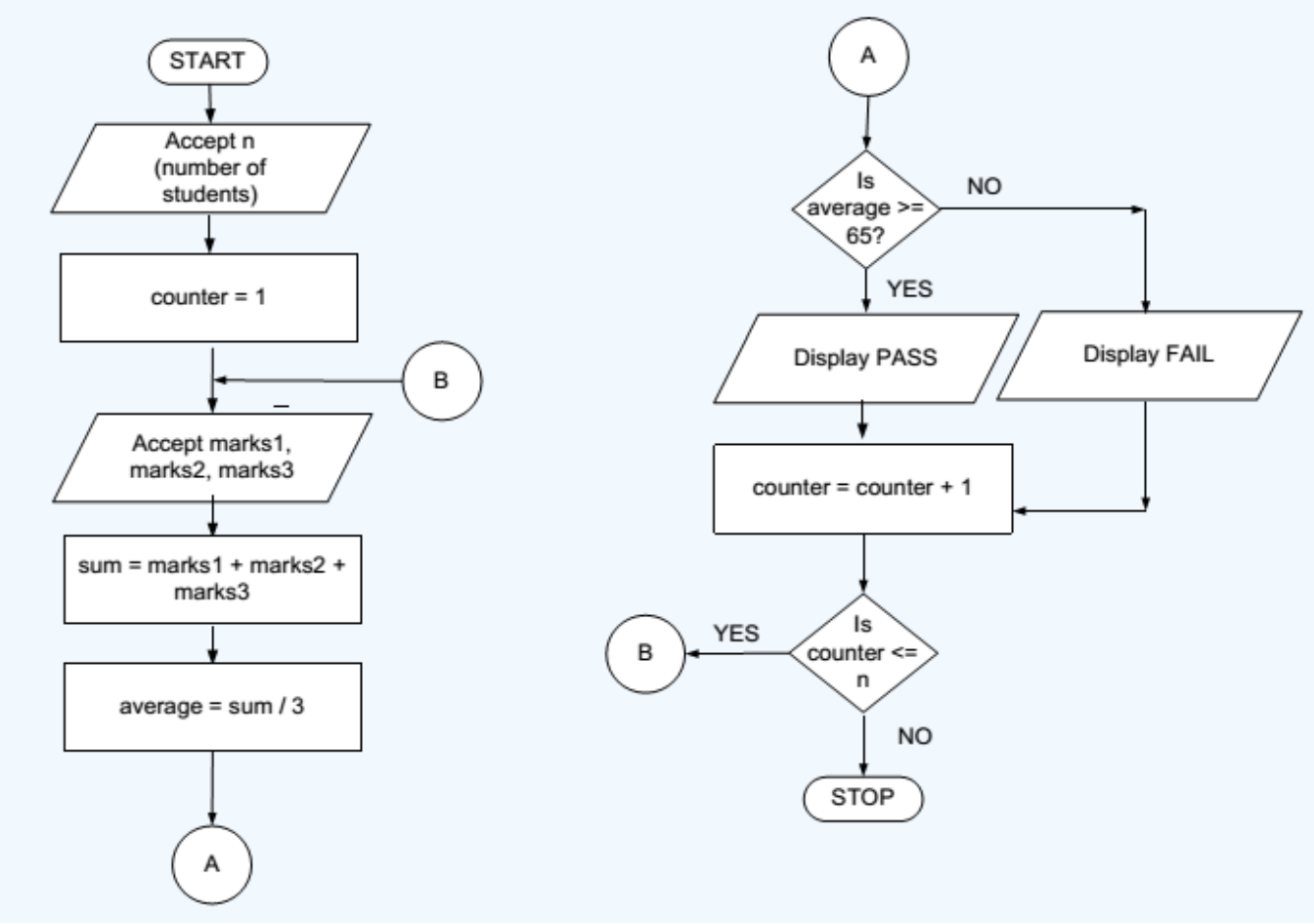


Flowchart for Nested Loop



Flowchart for Iterational Loop

- Write a program to find the average of marks scored by him in three subjects for 'N' students. And then test whether he passed or failed. For a student to pass, average should not be less than 65.



Tools to draw Flowcharts

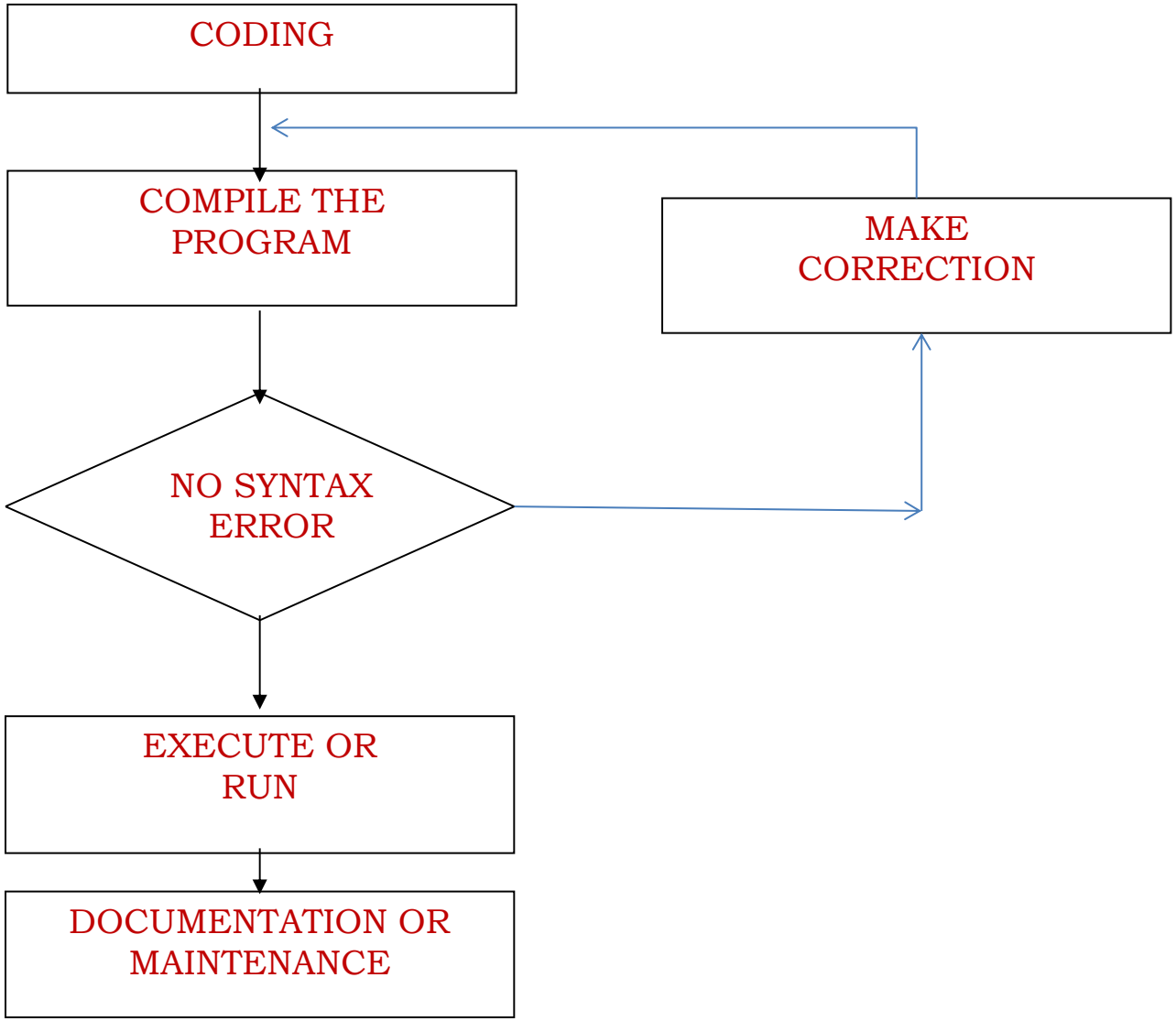
- Microsoft Visio
- Google Docs
- Gliffy Flowchart Software
- SmartDraw
- Creately
- Edraw Max
- Lucidchart
- Cacoo
- Flowchart.com
- yUML
- Diagramly
- **yEd**
- Graphviz
- SlickPlan
- Draw Anywhere
- Dia Diagram Editor
- SilverDiagram
- ArgoUML
- Allclear

These are some of the useful tools to create flow chart diagrams,

Programming or Implementation Phase

- Transcribing the logical flow of solution steps in flowchart or algorithm to program code and run the program code on a computer using a programming language.
- Programming phase takes 5 stages:
  - **Coding.**
  - **Compiling.**
  - **Debugging.**
  - **Run or Testing.**
  - **Documentation and maintenance.**
- Once the program is coded using one of the programming language, it will be compiled to ensure there is no syntax error.
- Syntax free program will then be executed to produce output and subsequently maintained and documented for later reference.

Programming or Implementation Phase



Programming or Implementation Phase

Coding :

- Translation or conversion of each operation in the flowchart or algorithm (pseudocode) into a computer-understandable language.
- Coding should follow the format of the chosen programming language.

Compiling and Debugging

- Compiling - Translates a program written in a particular high–level programming language into a form that the computer can understand.
- Compiler checks the program code so that any part of source code that does not follow the format or any other language requirements will be flagged as syntax error.
- This syntax error in also called bug, when error is found the programmer will debug or correct the error and then recompile the source code again
- Debugging process is continued until there is no more error in program(free from errors).

Programming or Implementation Phase

Testing :

- The program code that contains no more error is called executable program. It is ready to be tested.
- When it is tested, the data is given and the result is verified so that it should produced output as intended.
- Though the program is error free, sometimes it does not produced the right result. In this case the program faces **logic error**.
- Incorrect sequence of instruction is an example that causes **logic error**.

Programming or Implementation Phase

Documentation and Maintenance

- When the program is thoroughly tested for a substantial period of time and it is consistently producing the right output, it can be documented.
- Documentation is important for future reference. Other programmer may take over the operation of the program and the best way to understand a program is by studying the documentation.
- Trying to understand the logic of the program by looking at the source code is not a good approach.
- Studying the documentation is necessary when the program is subjected to enhancement or modification.
- Documentation is also necessary for management use as well as audit purposes.

Best Practices

Develop efficient computer solution to problems:

1. Use Modules

   Use four logic structures

   a. Sequential structure
      - Executes instructions one after another in a sequence.

   b. Decision structure
      - Branches to execute one of two possible sets of instructions.

   c. Loop structure
      - Executes set of instruction many times.

   d. Case structure
      - Executes one set of instructions out of several sets.

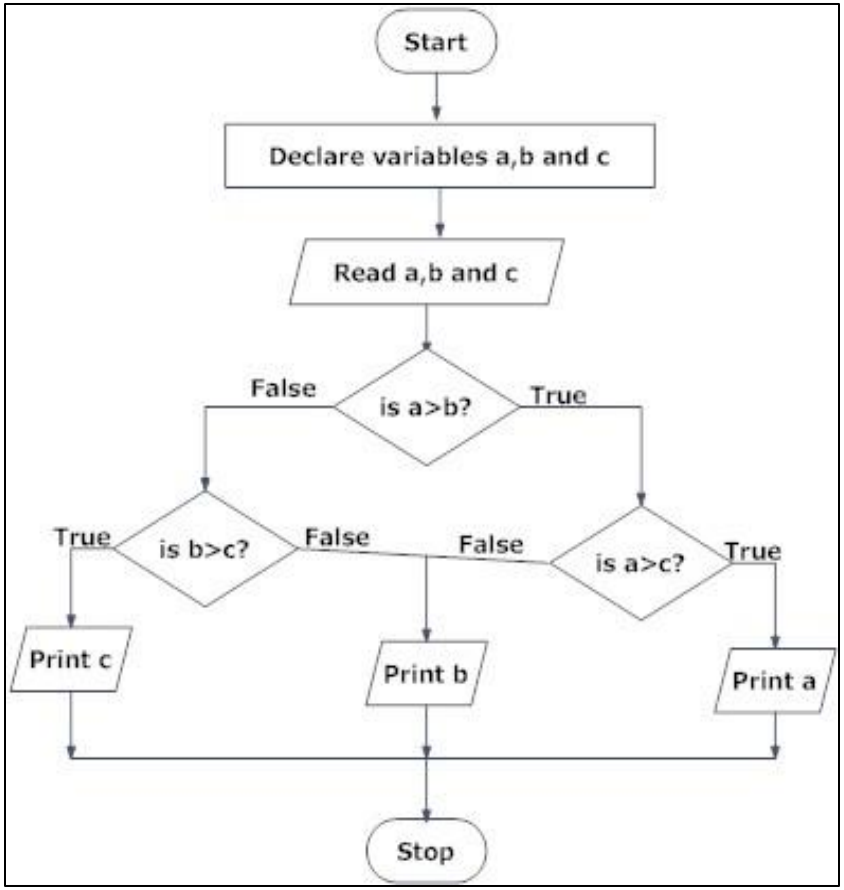
2. Eliminate rewriting of identical process by using modules.

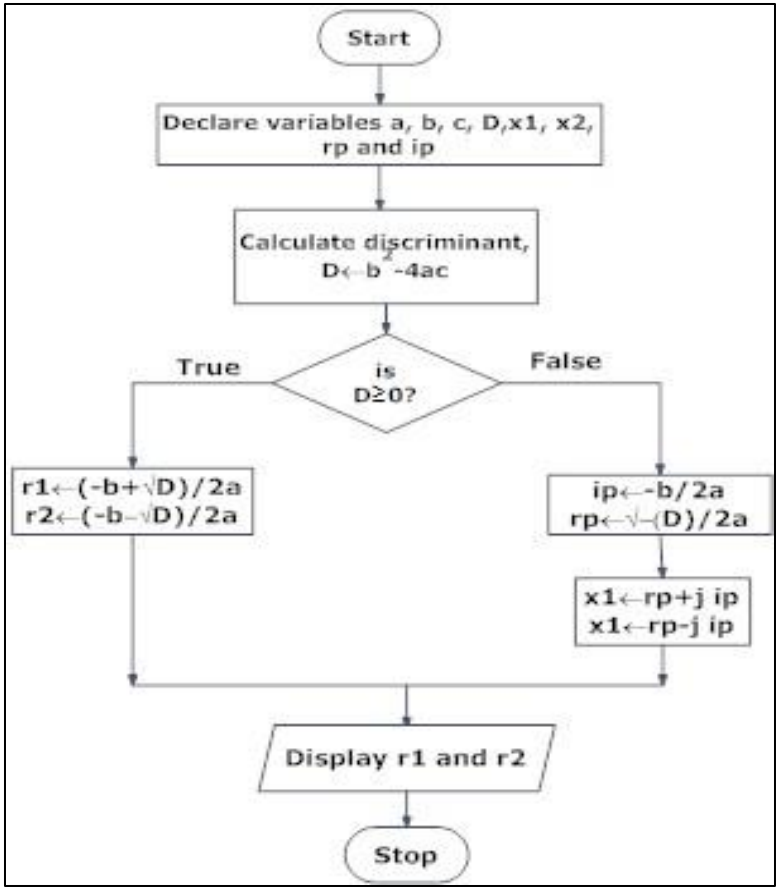
3. Use techniques to improve readability including four logic structure, proper naming of variables, internal documentation and proper indentation.

Practice Problem

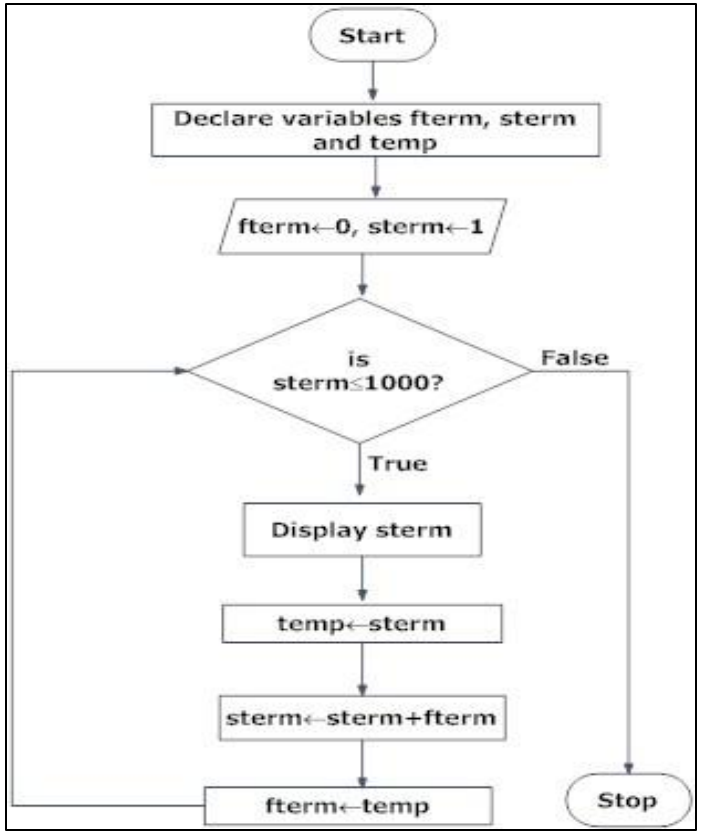
- Draw the Flow chart for the following :
  - Largest of 2 Numbers.
  - Largest of 3 Numbers.
  - Roots of a Quadratic Equation $\mathbf{Ax^2 + Bx + C}$.
  - To perform arithmetic operations.
  - Display Temperature.

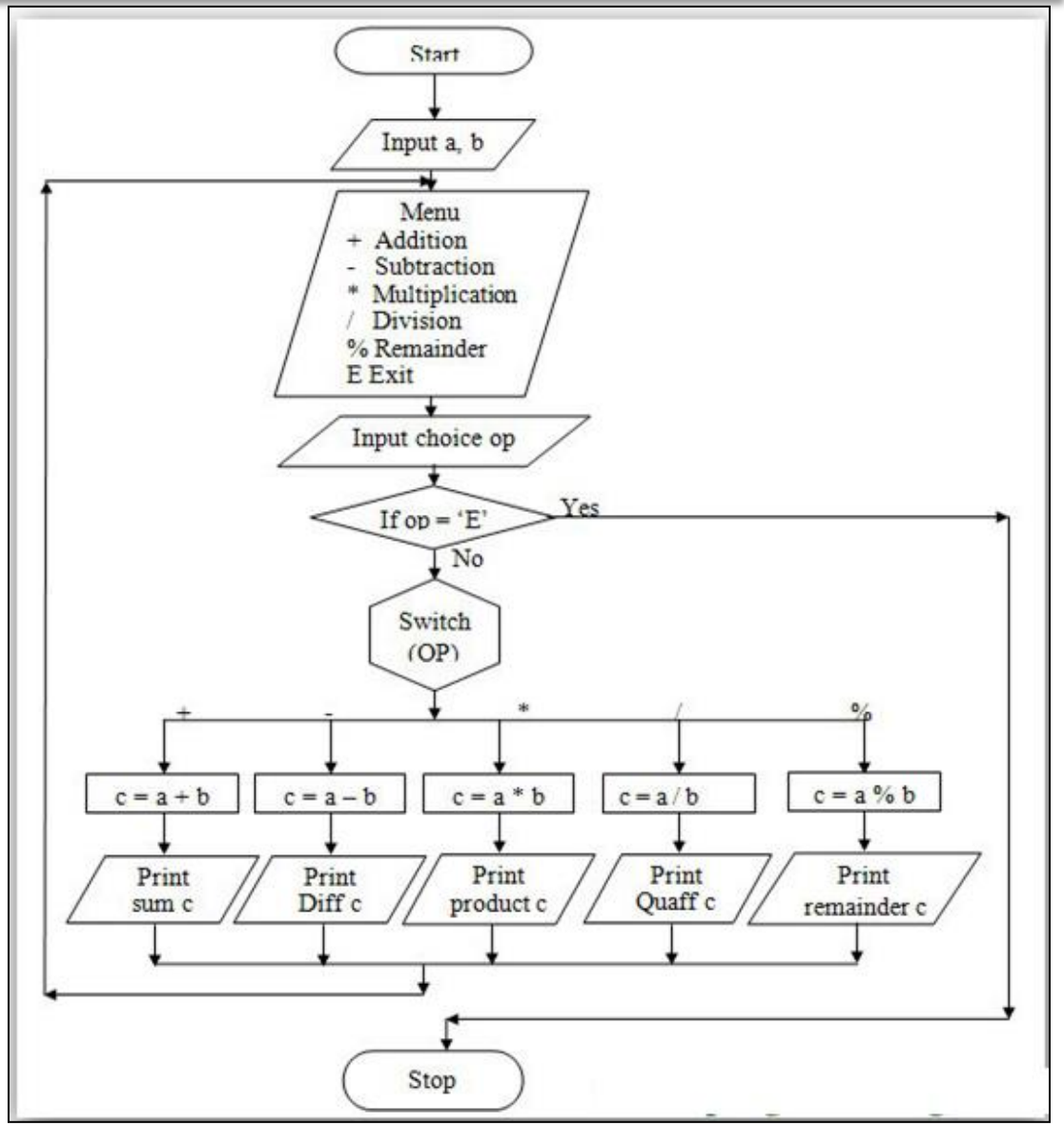
Examples


Start
Declare variables a,b and c
Read a,b and c
is a>b?
False
True
is b>c?
True
False
is a>c?
False
True
Print c
Print b
Print a
Stop


Start
Declare variables a, b, c, D,x1, x2, rp and ip
Calculate discriminant, D←b²-4ac
is D≥0?
True
False
r1←(-b+√D)/2a
r2←(-b-√D)/2a
ip←-b/2a
rp←√(D)/2a
x1←rp+j ip
x1←rp-j ip
Display r1 and r2
Stop

Examples


Start
Declare variables fterm, sterm and temp
fterm←0, sterm←1
is sterm≤1000?
False
True
Display sterm
temp←sterm
sterm←sterm+fterm
fterm←temp
Stop


Start
Input a, b
Menu
+ Addition
- Subtraction
* Multiplication
/ Division
% Remainder
E Exit
Input choice op
If op = 'E'
Yes
No
Switch (OP)
+
-
*
/
%
c = a + b
c = a – b
c = a * b
c = a / b
c = a % b
Print sum c
Print Diff c
Print product c
Print Quaff c
Print remainder c
Stop


Happy
Learning