Pseudocode

Pseudocode

- Pseudocode is a kind of **structured English for describing algorithms.**
- It allows the designer to focus on the logic of the algorithm without being distracted by deta ils of language syntax.
- Pseudocode needs to be **complete**.
- It describe the entire logic of the algorithm so that implementation becomes a rote mechanical task of transla ting line by line into source code.

Keywords in Pseudocode

- Several keywords are often used to **indicate common** in put, output, and **processing operations**.

- **Input:** READ, OBTAIN, GET
- **Output:** PRINT, DISPLAY, SHOW
- **Compute:** COMPUTE, CALCULATE, DETERMINE
- **Initialize:** SET, INIT
- **Add one:** INCREMENT, BUMP

Pseudocode for Sequential Problem

Find area of rectangle

Pseudocode for Sequential Problem

Find area of rectangle

- READ height of rectangle
- READ width of rectangle
- COMPUTE area as height times width

IF THEN ELSE

- Binary choice on a given **Boolean condition** is indicated by the use of **four keywords**:
- **IF, THEN, ELSE**, and **ENDIF**.

The general form is:

```
IF condition THEN
        sequence 1
ELSE
        sequence 2
ENDIF
```

Example

```
IF HoursWorked > NormalMax THEN
        Display overtime message
ELSE
        Display regular time message
ENDIF
```

Problem

Write Pseudocode for Sum of 1 to 100

WHILE

- used to specify a loop with a test at the top.
- beginning and ending of the loop are indicated by **two keywords: WHILE** and **ENDWHILE.**
- **General form is:**

```
WHILE condition
    sequence
ENDWHILE
```

WHILE

- Loop is entered only if the condition is true.
- "sequence" is performed for each iteration.
- At the conclusion of each iteration, the condition is evaluated and the loop continues as lon g as the condition is true.

Example

```
WHILE Population < Limit
        Compute Population as Population + Births - Deaths
ENDWHILE
WHILE employee.type NOT EQUAL manager AND
                                personCount < numEmployees
INCREMENT personCount
CALL employeeList.getPerson with personCount
                                    RETURNING employee
ENDWHILE
```

FOR Loop

- This loop is a specialized construct for iterating a specific number of times, often called a "counting" loop.
- **Two keywords**: **FOR** and **ENDFOR** are used.
- **The general form is:**

```
FOR iteration bounds
        sequence
ENDFOR
```

Example FOR Loop

- FOR each month of the year (good)
- FOR month = 1 to 12 (ok)
- FOR each employee in the list (good)
- FOR empno = 1 to listsize (ok)

Problem

Write Pseudocode to find average marks scored by ‘N’ students

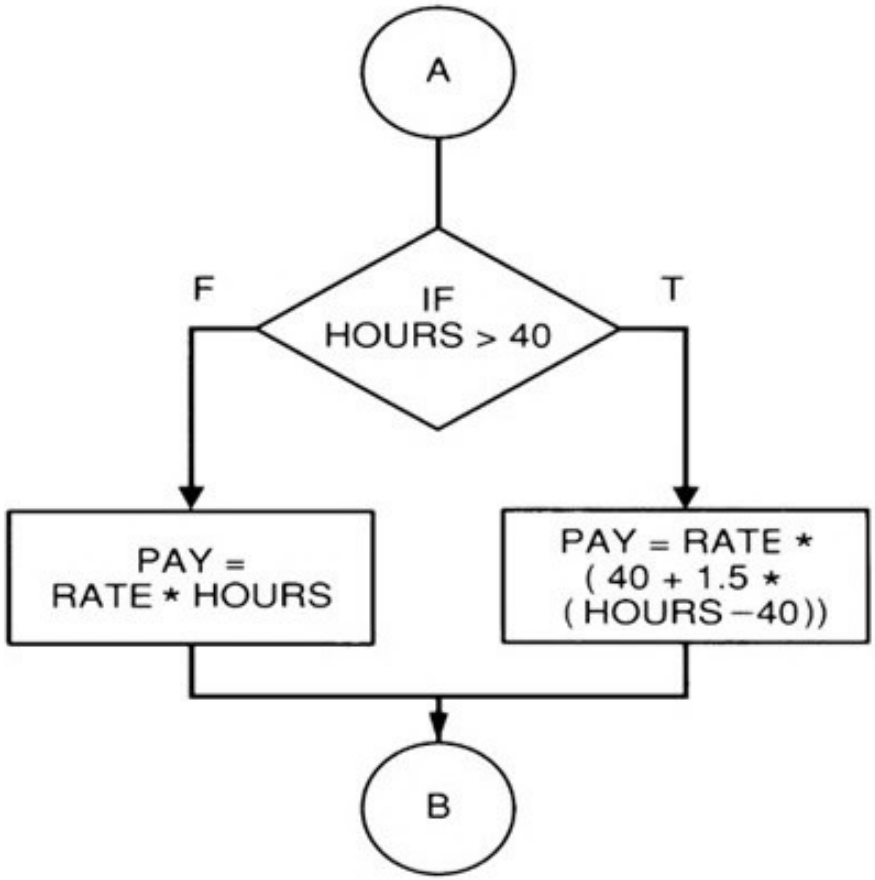
Nested Constructs

```
SET total to zero
REPEAT
        READ Temperature
        IF Temperature > Freezing THEN
                        INCREMENT total
        END IF
UNTIL Temperature < zero
Print total
```

Problem

- 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.

Example Decision Structure

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

INVOKING SUBPROCEDURES

- Use the **CALL** keyword.

For example:

CALL AvgAge with StudentAges

CALL Swap with CurrentItem and TargetItem

CALL Account.debit with CheckAmount

CALL getBalance RETURNING aBalance

CALL SquareRoot with orbitHeight
RETURNING nominalOrbit

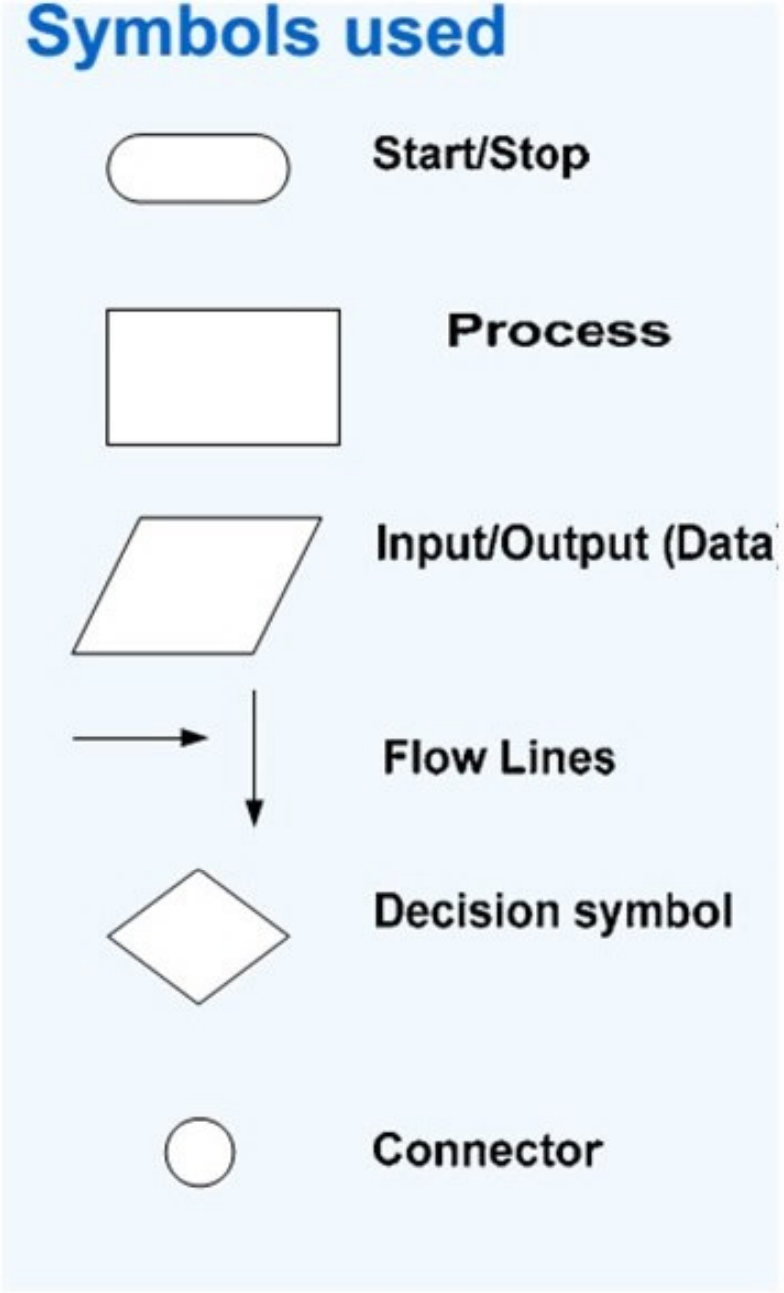
Flow Chart and Phases of Making an Executable Code

Drawing Flowcharts

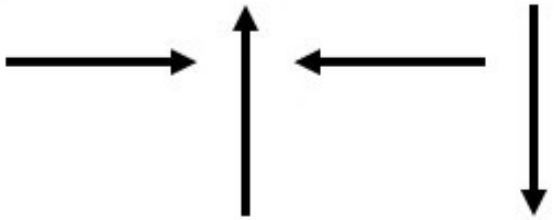
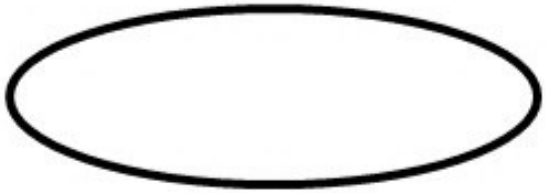
- Flowchart is the graphic representations of the individual steps or actions to implement a particular module
- Flowchart can be likened to the blueprint of a building
- An architect draws a blueprint before beginning construction on a building, so the programmer draws a flowchart before writing a program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

Flow Charts

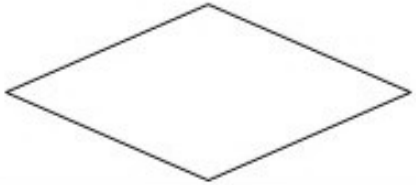
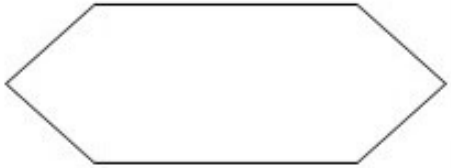
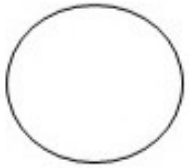
A flow chart is an organized combination of shapes, lines and text that graphically illustrate a process or structure.



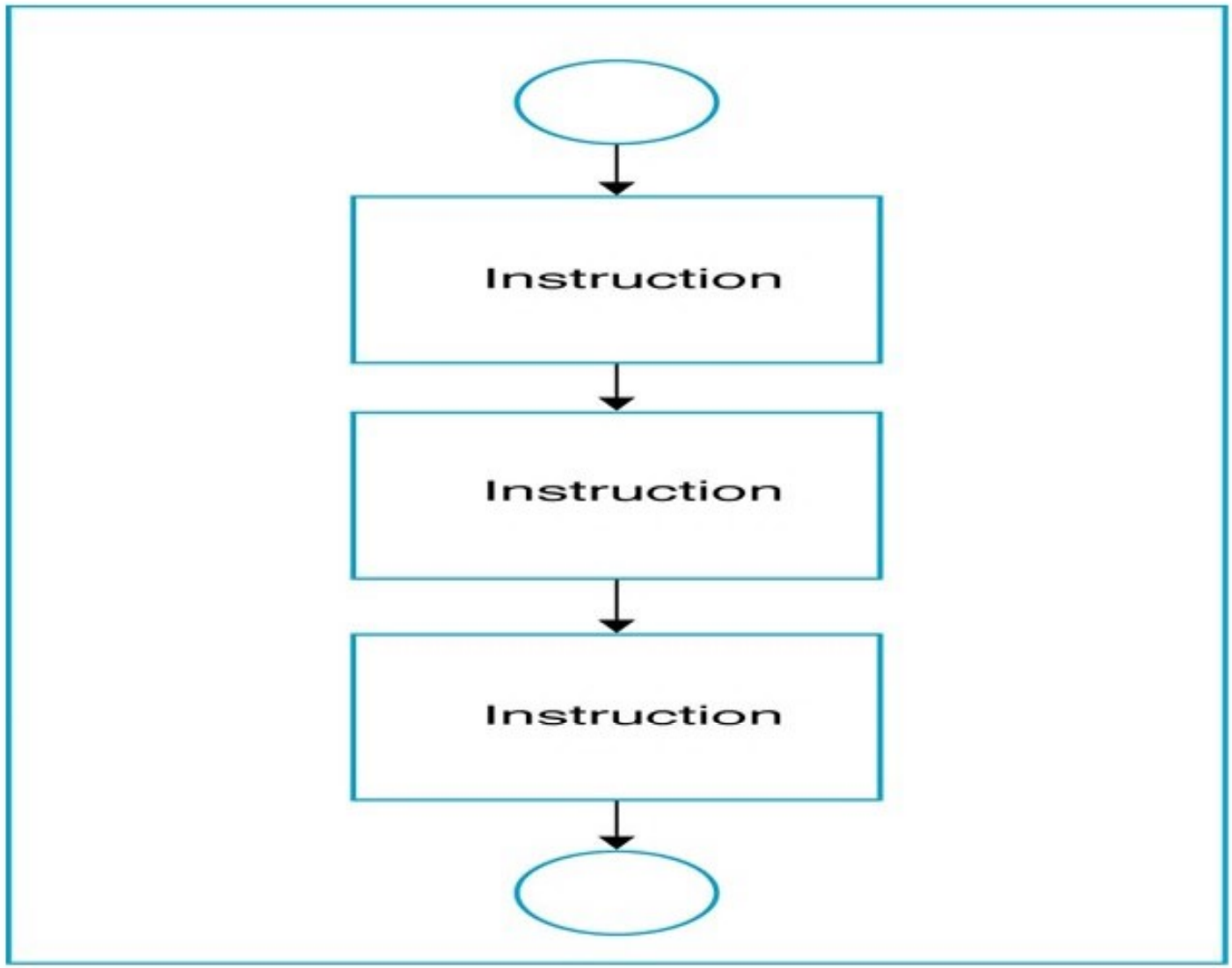
PRE-PROGRAMMING PHASE

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.

PRE-PROGRAMMING PHASE

	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 Logic Structure



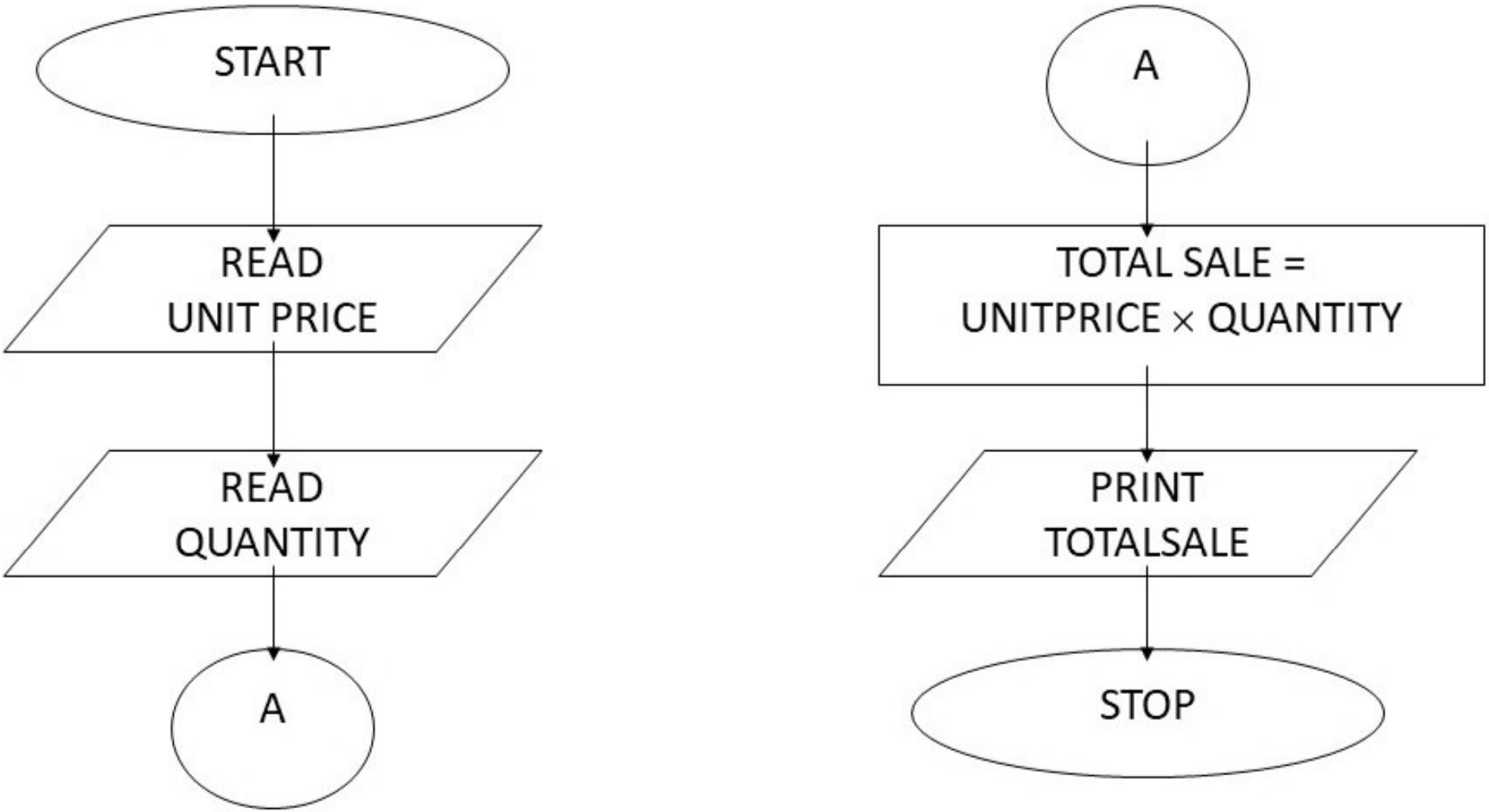
Sale Problem

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

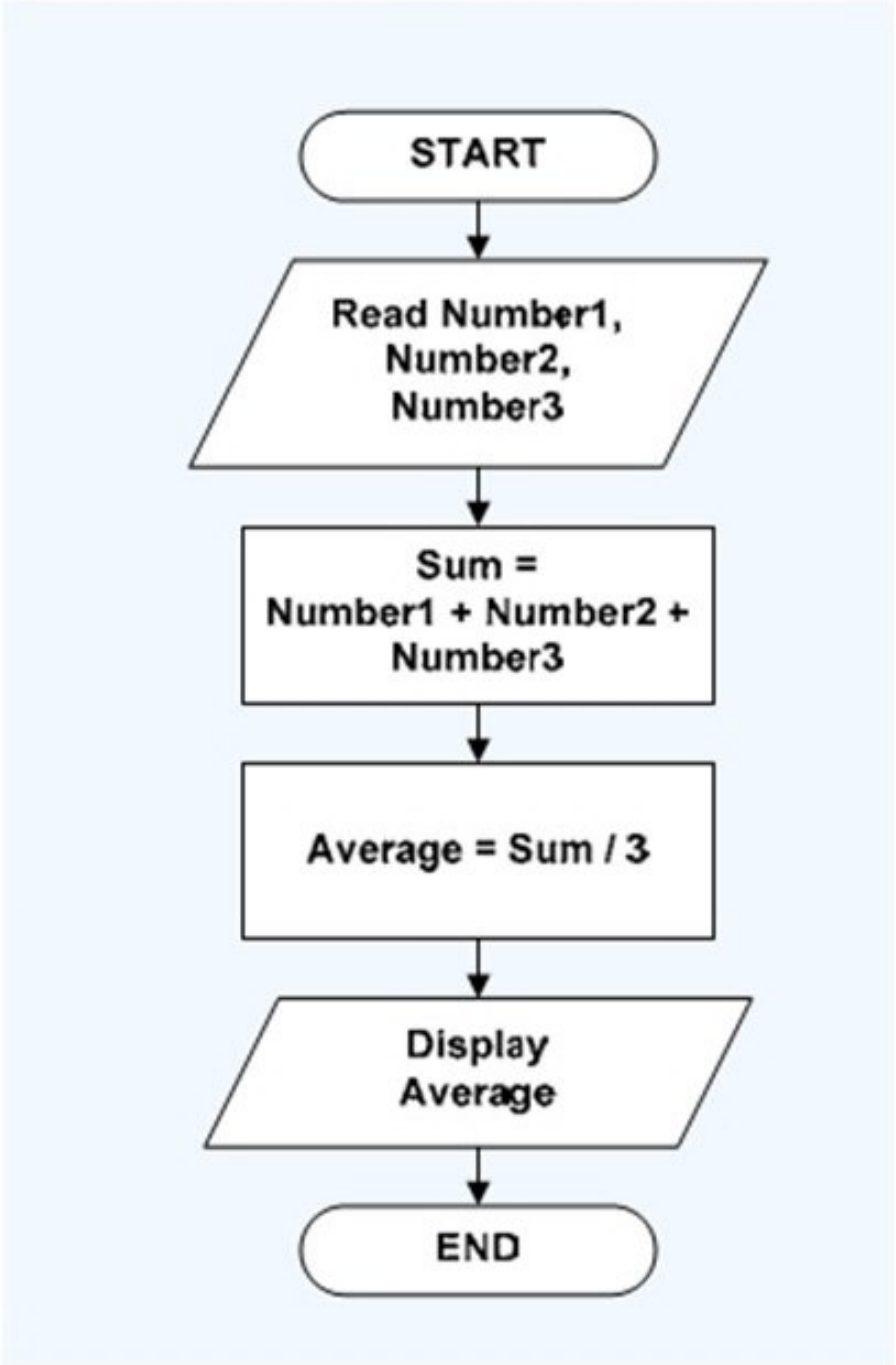
Solution: Stepwise Analysis of the Sale Problem

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

PRE-PROGRAMMING PHASE



Find the average of three numbers



The Decision Logic Structure

- 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:

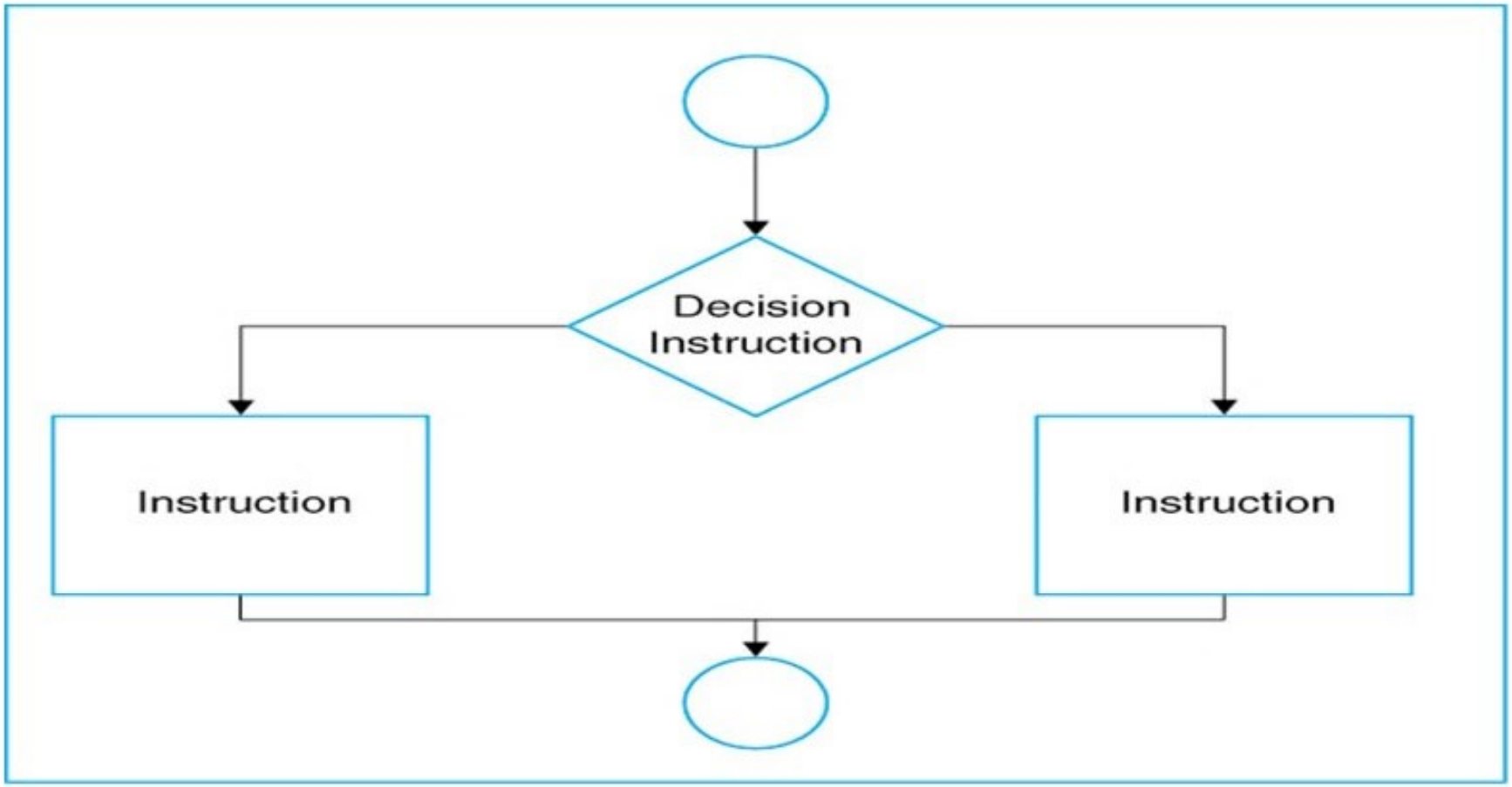
IF <condition(s)> THEN

<TRUE instruction(s)>

ELSE

<FALSE instruction(s)

Decision Logic Structure



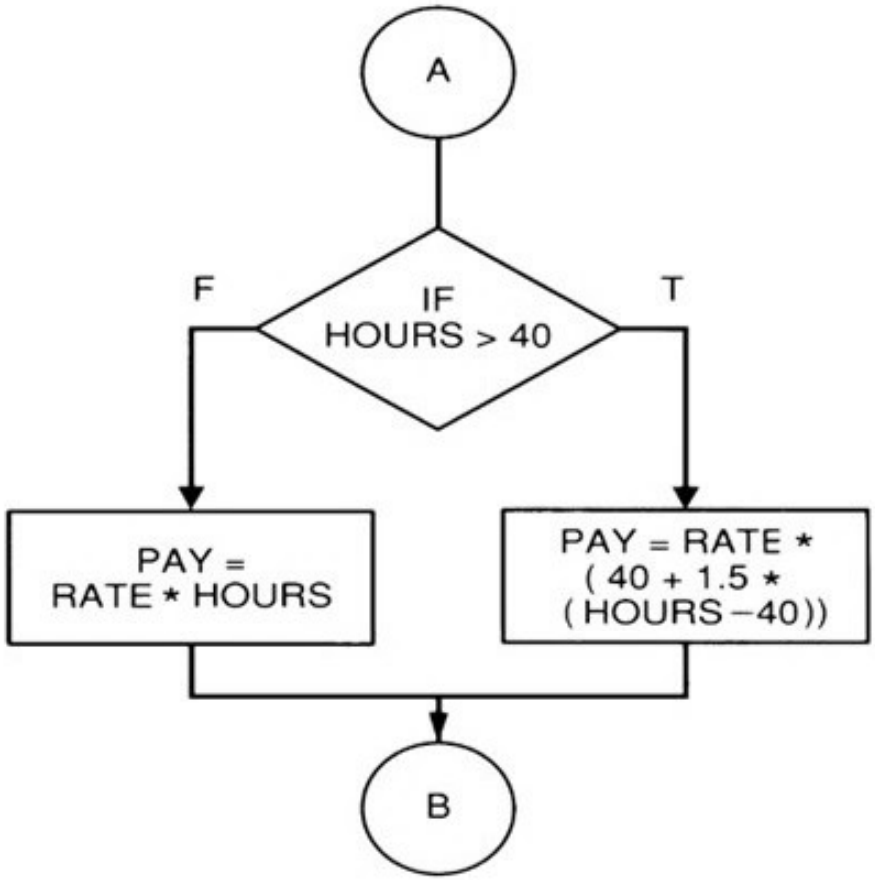
Examples of conditional expressions

- 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)

Conditional Pay Calculation

- 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.

Example Decision Structure

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

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.

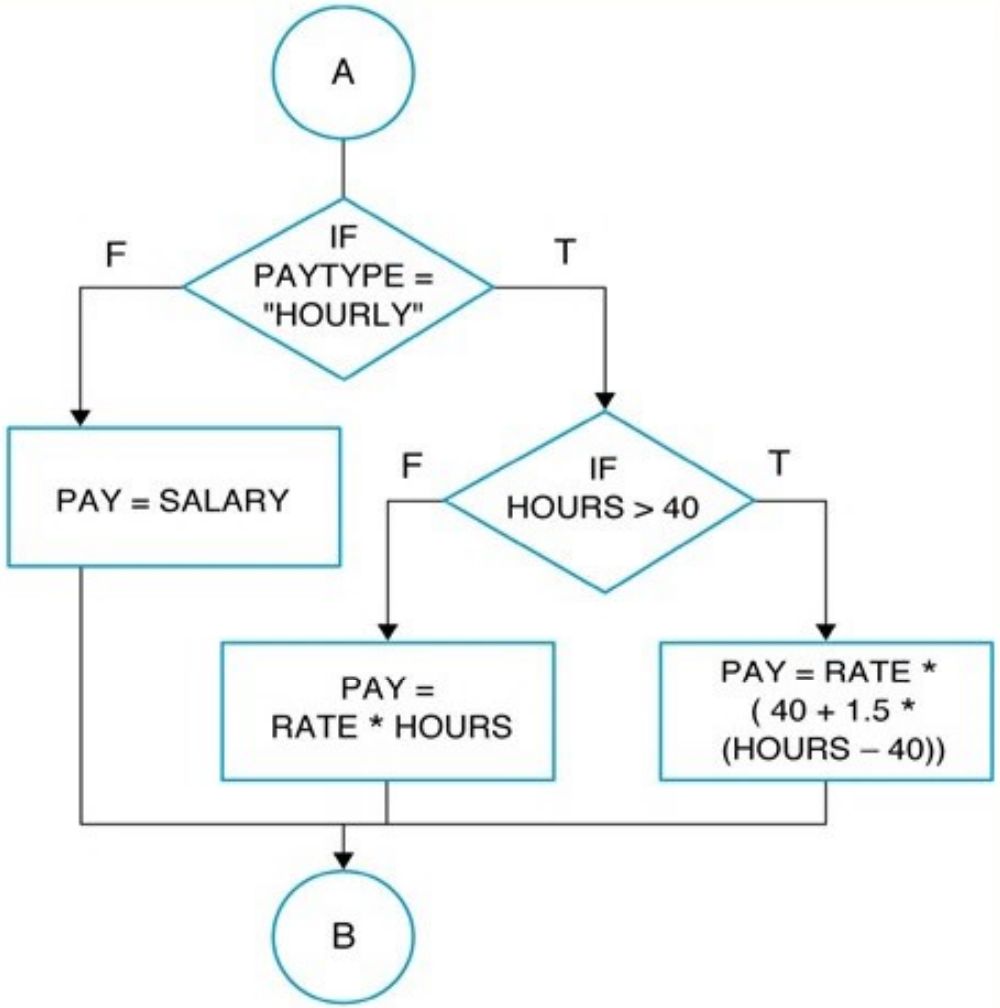
NESTED IF/THEN/ELSE INSTRUCTIONS

Algorithm

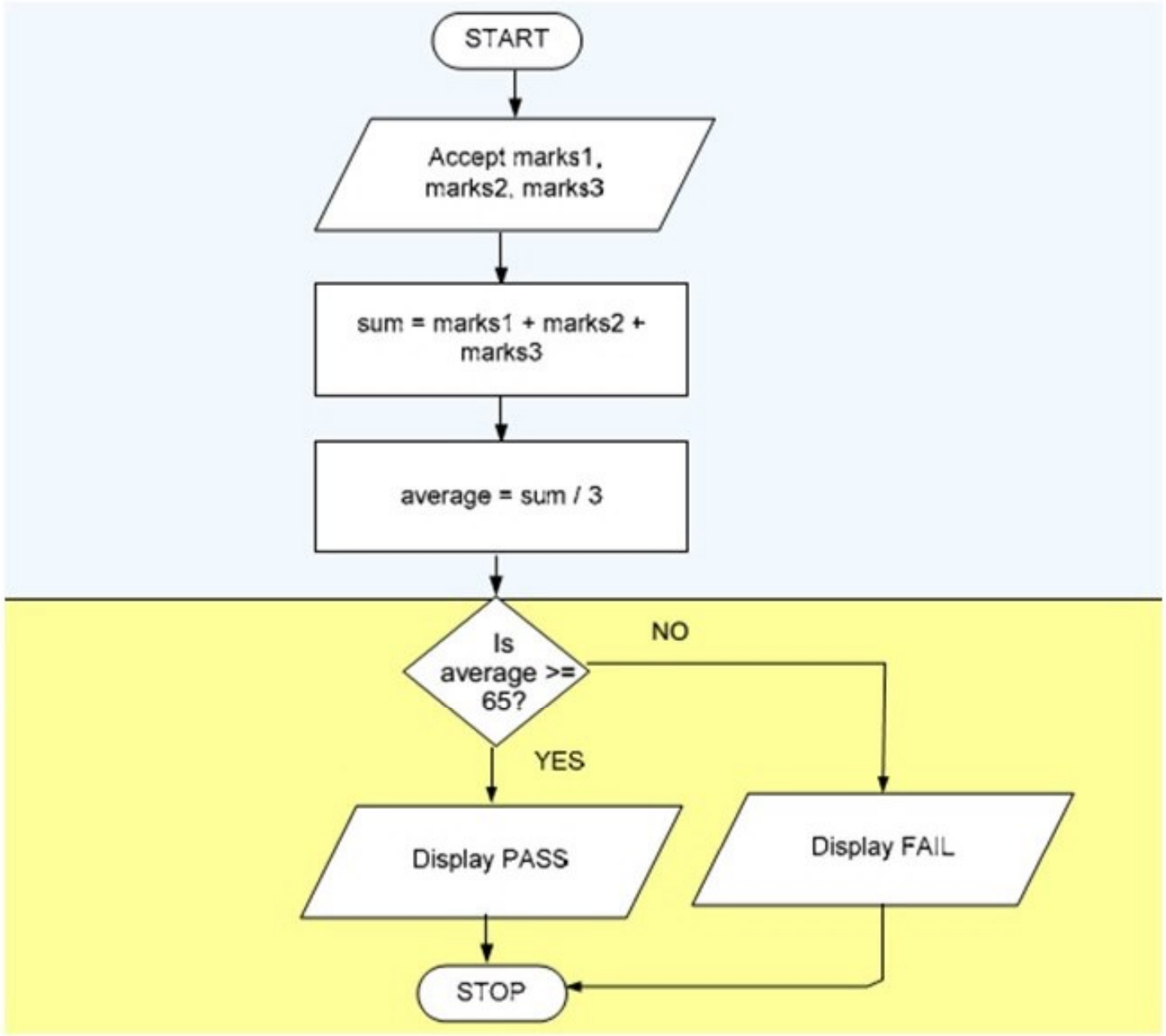
```
IF PAYTYPE = "HOURLY"
THEN
    IF HOURS > 40
    THEN
        PAY = RATE * (40
          + 1.5 * (HOURS
          – 40))
    ELSE
        PAY = RATE * HOURS
ELSE
    PAY = SALARY
```

T T F F

Flowchart



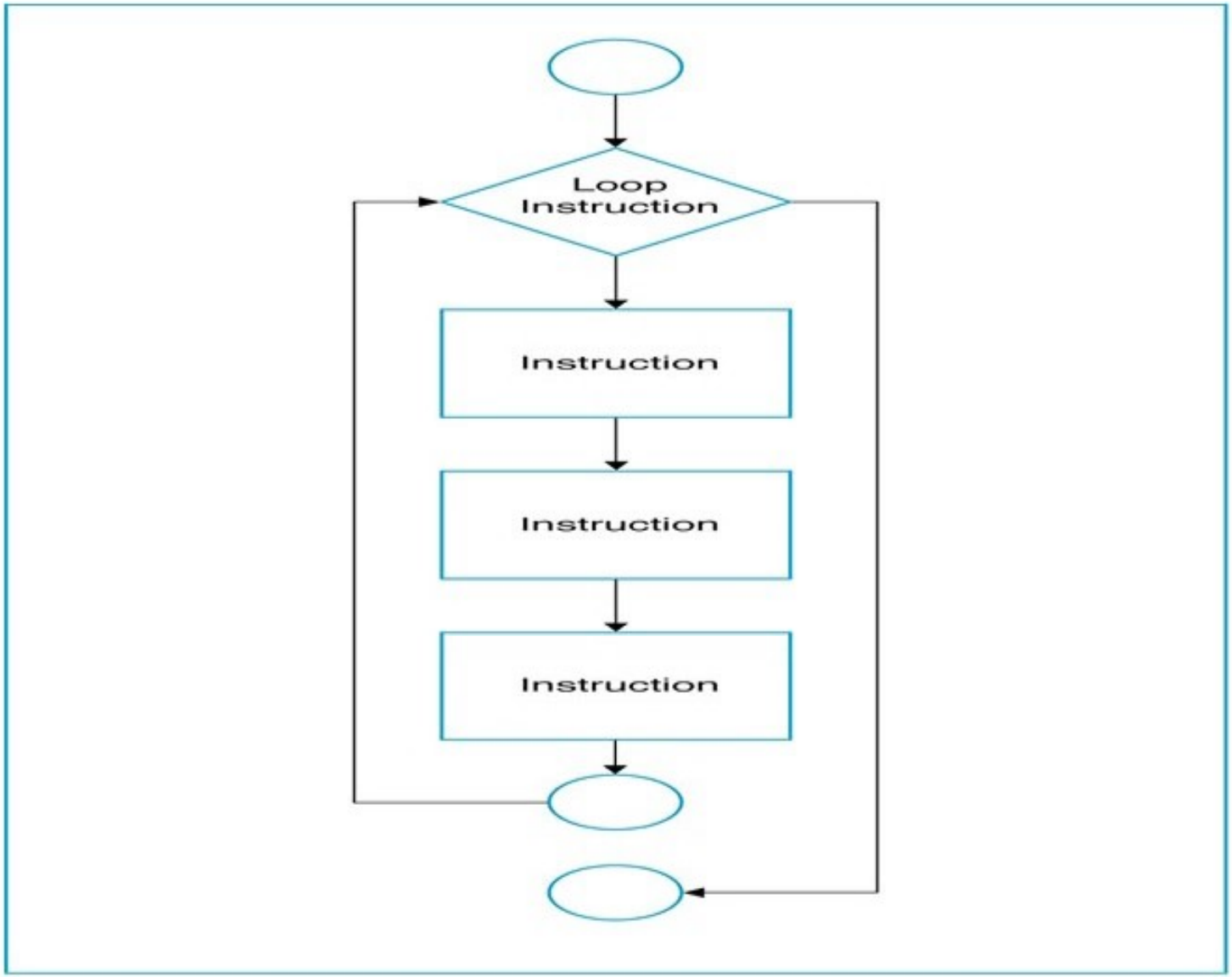
Flow Chart - Selectional



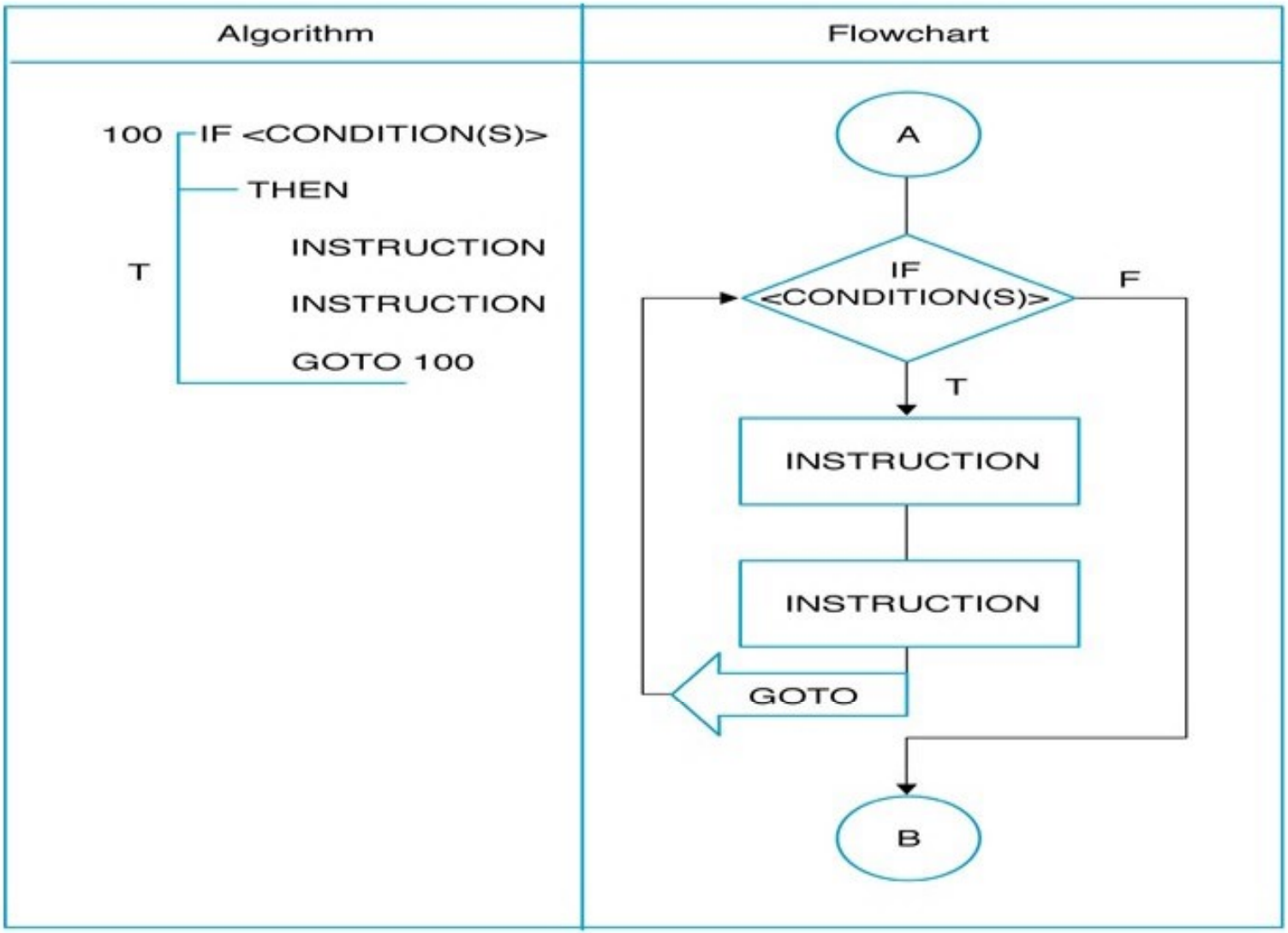
Iterational Structure

- 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

Loop Logic Structure



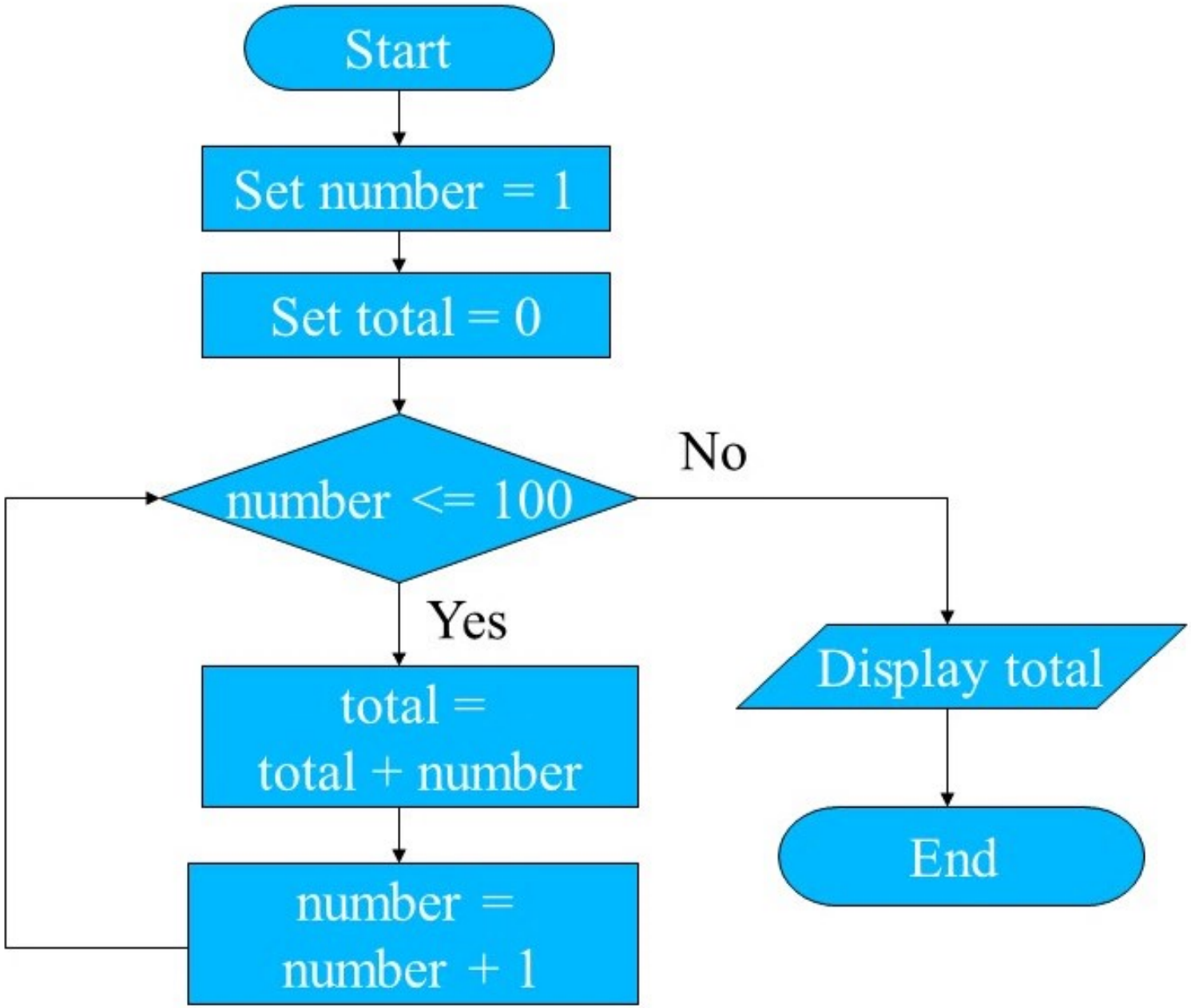
WHILE loop


Algorithm
Flowchart
100 IF <CONDITION(S)>
THEN
T
INSTRUCTION
INSTRUCTION
GOTO 100
A
IF
<CONDITION(S)>
F
T
INSTRUCTION
INSTRUCTION
GOTO
B

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

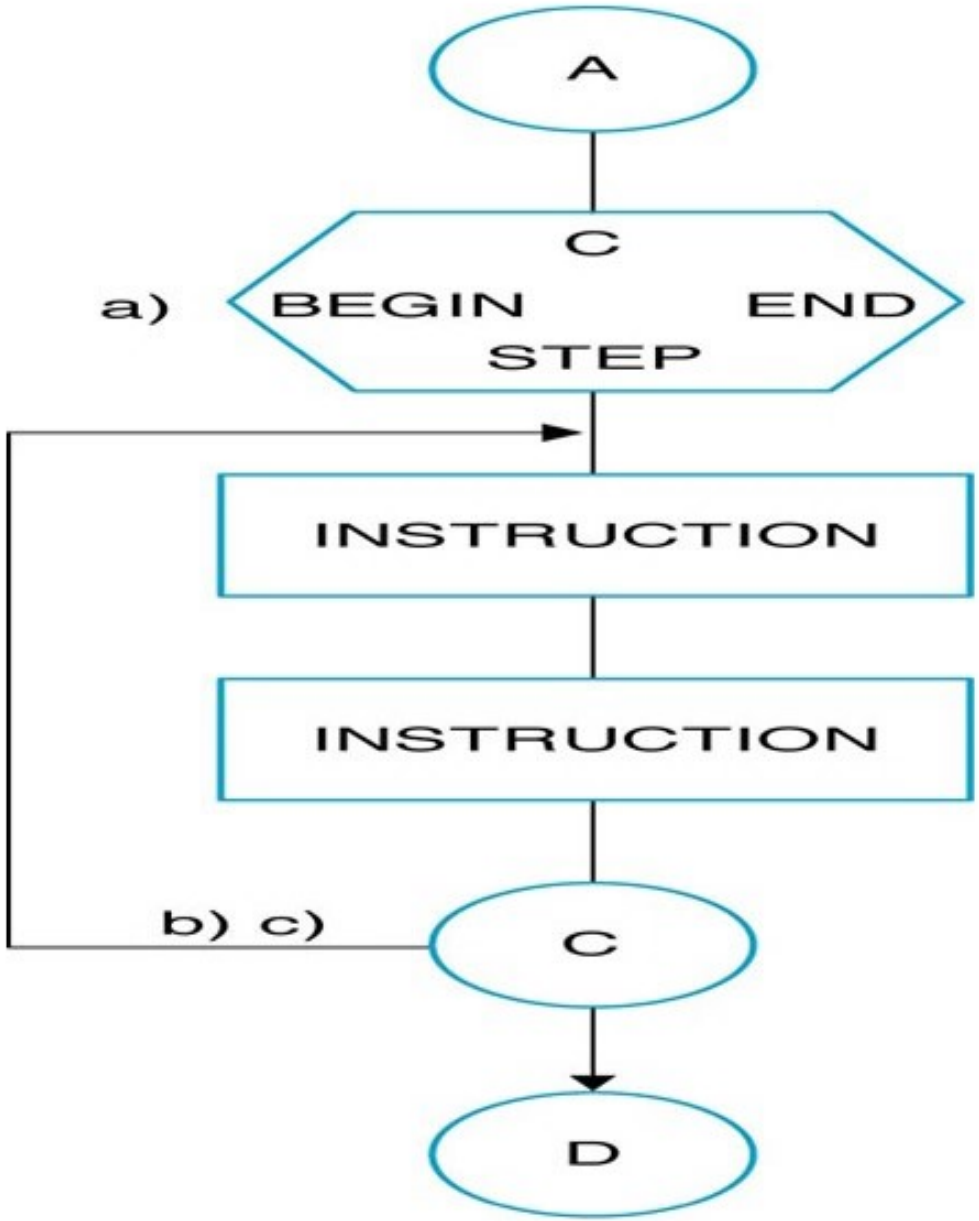
WHILE loop


Start
Set number = 1
Set total = 0
number <= 100
No
Yes
total =
total + number
number =
number + 1
Display total
End

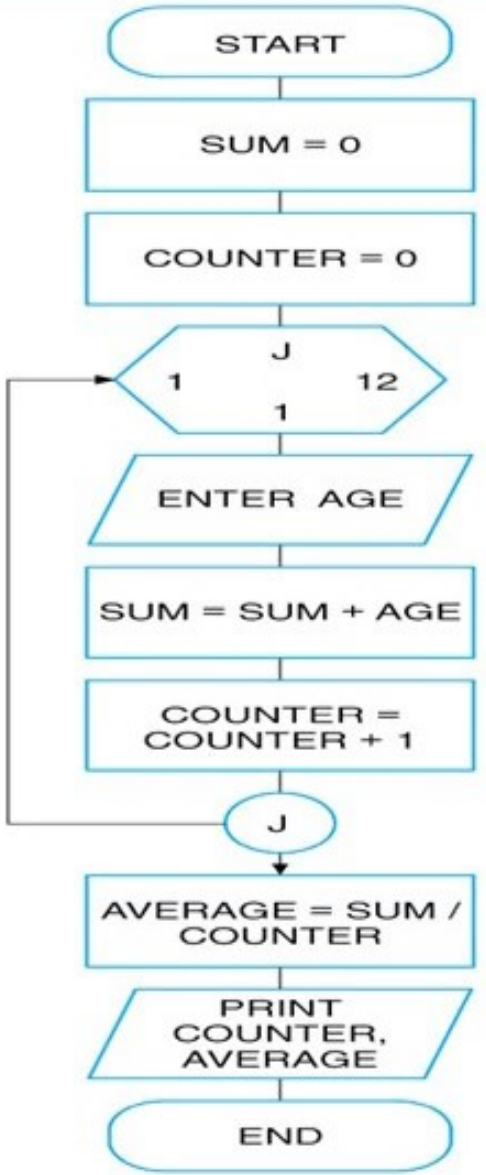
Automatic Counter Loop

- 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.

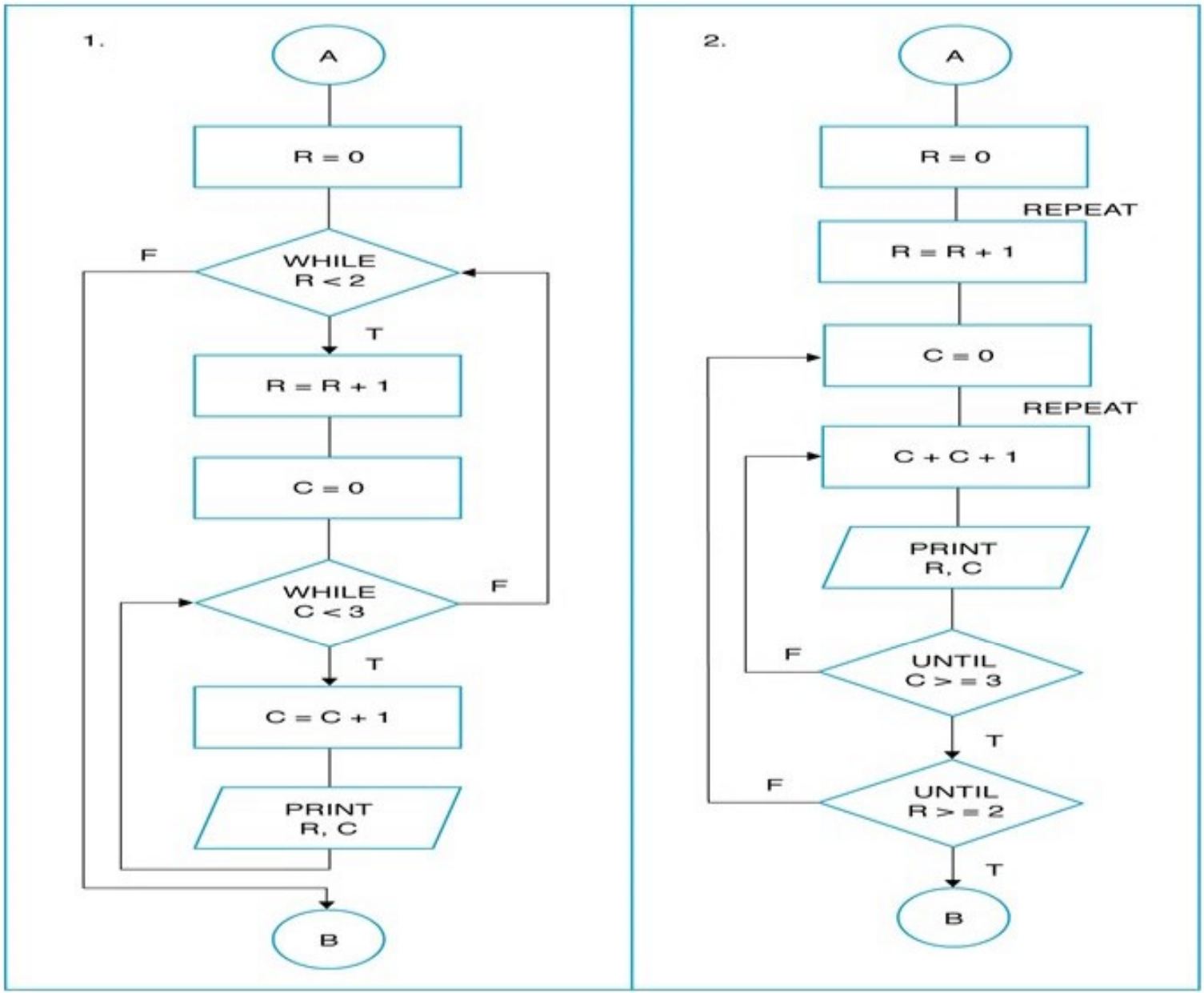
Automatic-Counter Loop



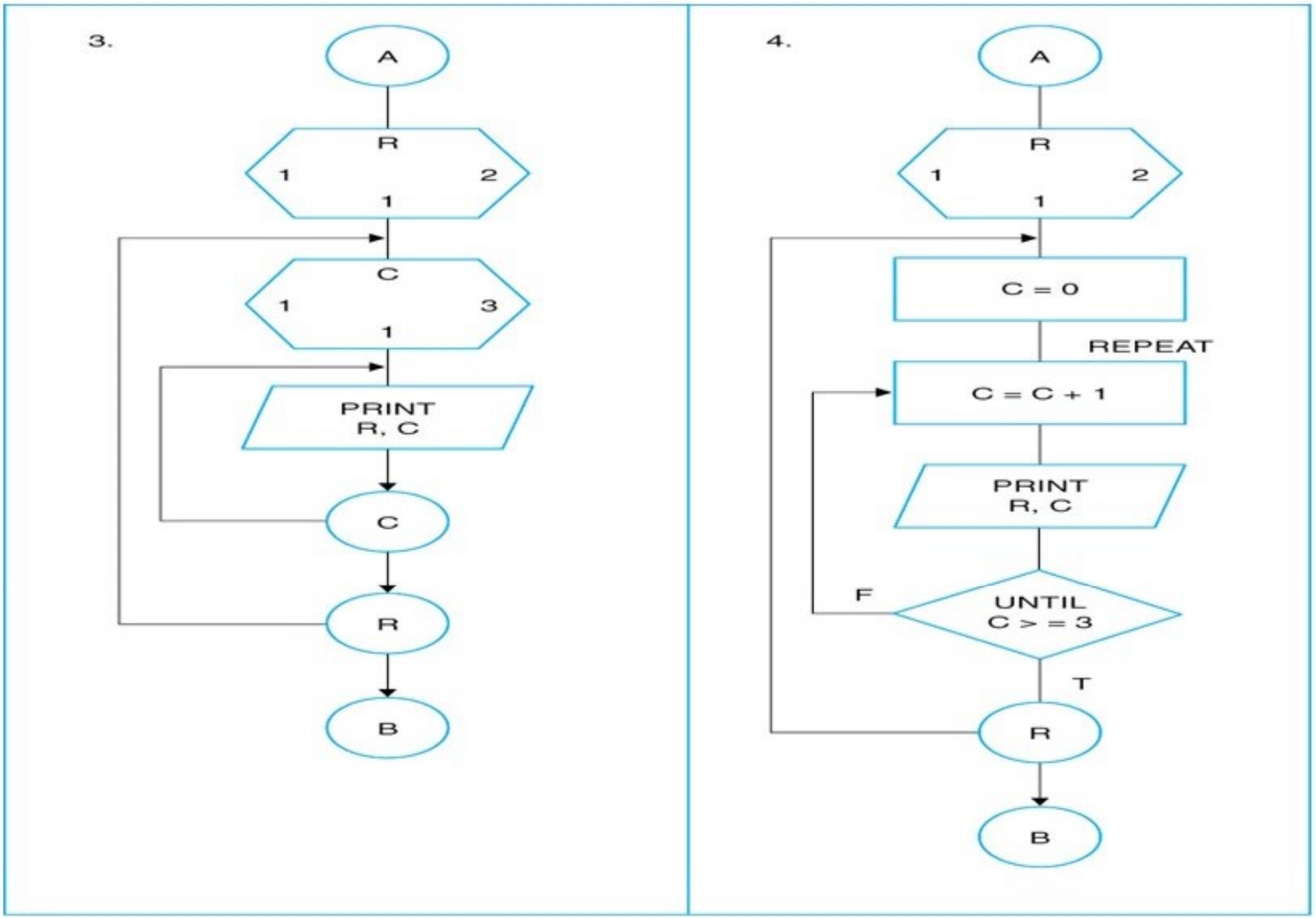
Automatic-Counter Loop

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	

NESTED LOOP


1.
A
R = 0
WHILE
R < 2
F
T
R = R + 1
C = 0
WHILE
C < 3
F
T
C = C + 1
PRINT
R, C
B
2.
A
R = 0
REPEAT
R = R + 1
C = 0
REPEAT
C + C + 1
PRINT
R, C
F
UNTIL
C > = 3
T
F
UNTIL
R > = 2
T
B

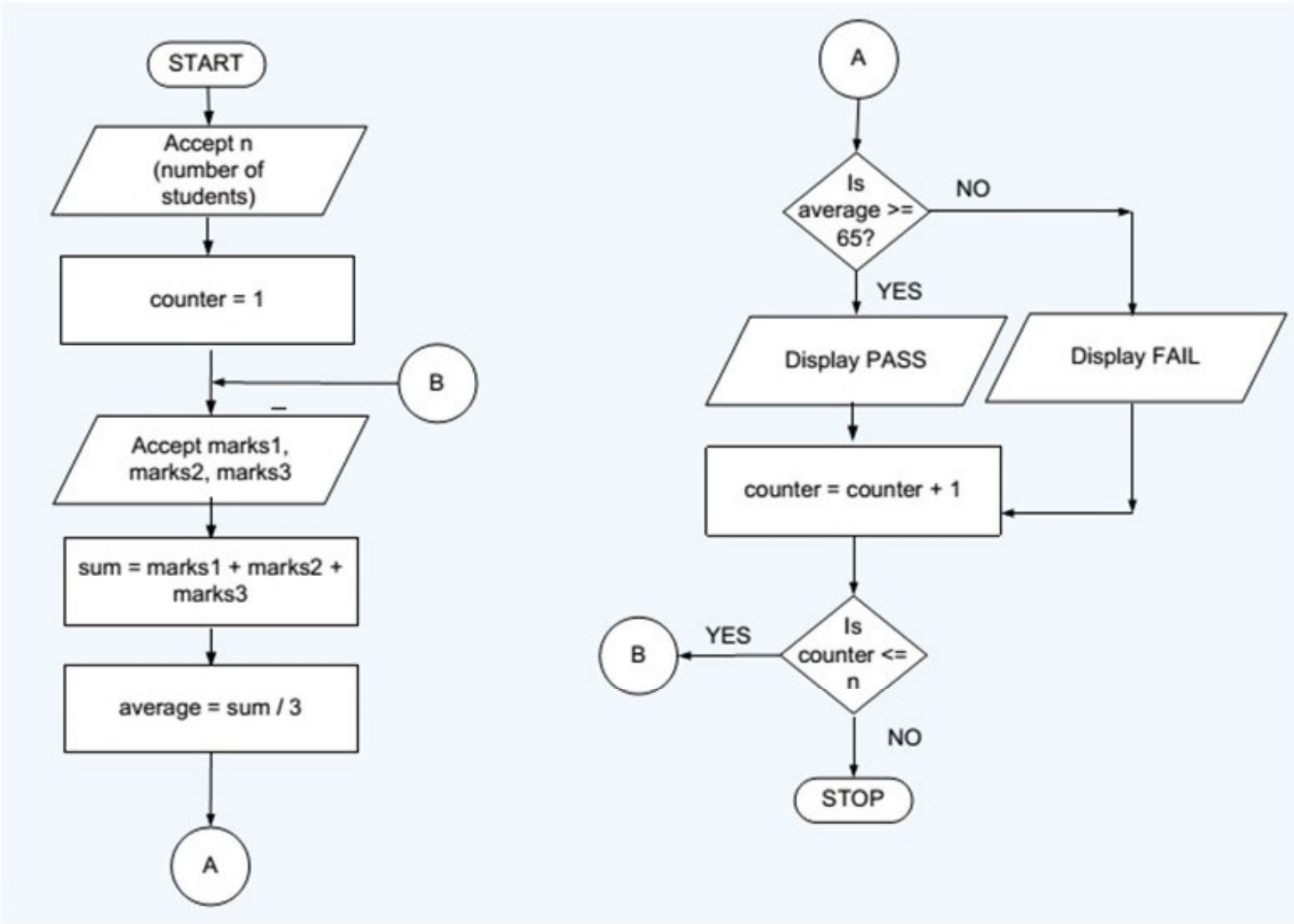
NESTED LOOP


3.
A
R
1
2
1
C
1
3
1
PRINT
R, C
C
R
B
4.
A
R
1
2
1
C = 0
REPEAT
C = C + 1
PRINT
R, C
F
UNTIL
C > = 3
T
R
B

Example (Iterational)

- 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.

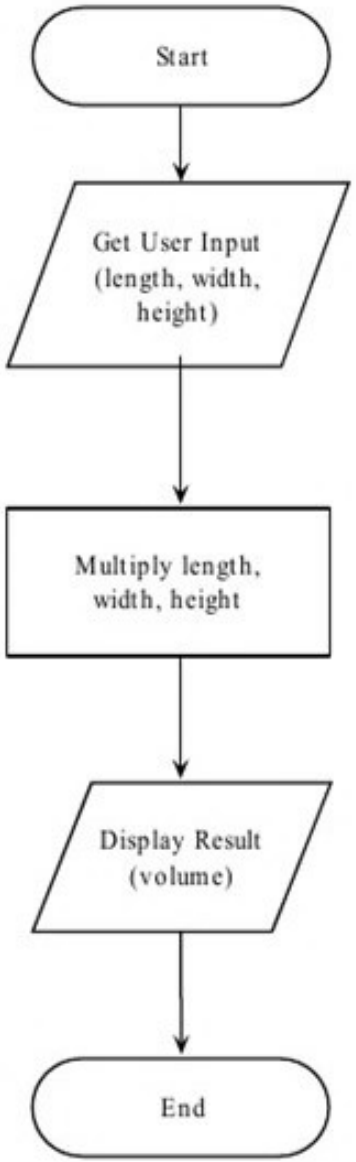
Flow Chart Iterational



Tool demo

- Yed tool shall be used for giving demo

Pseudocode – Partial English and Programming Language terms



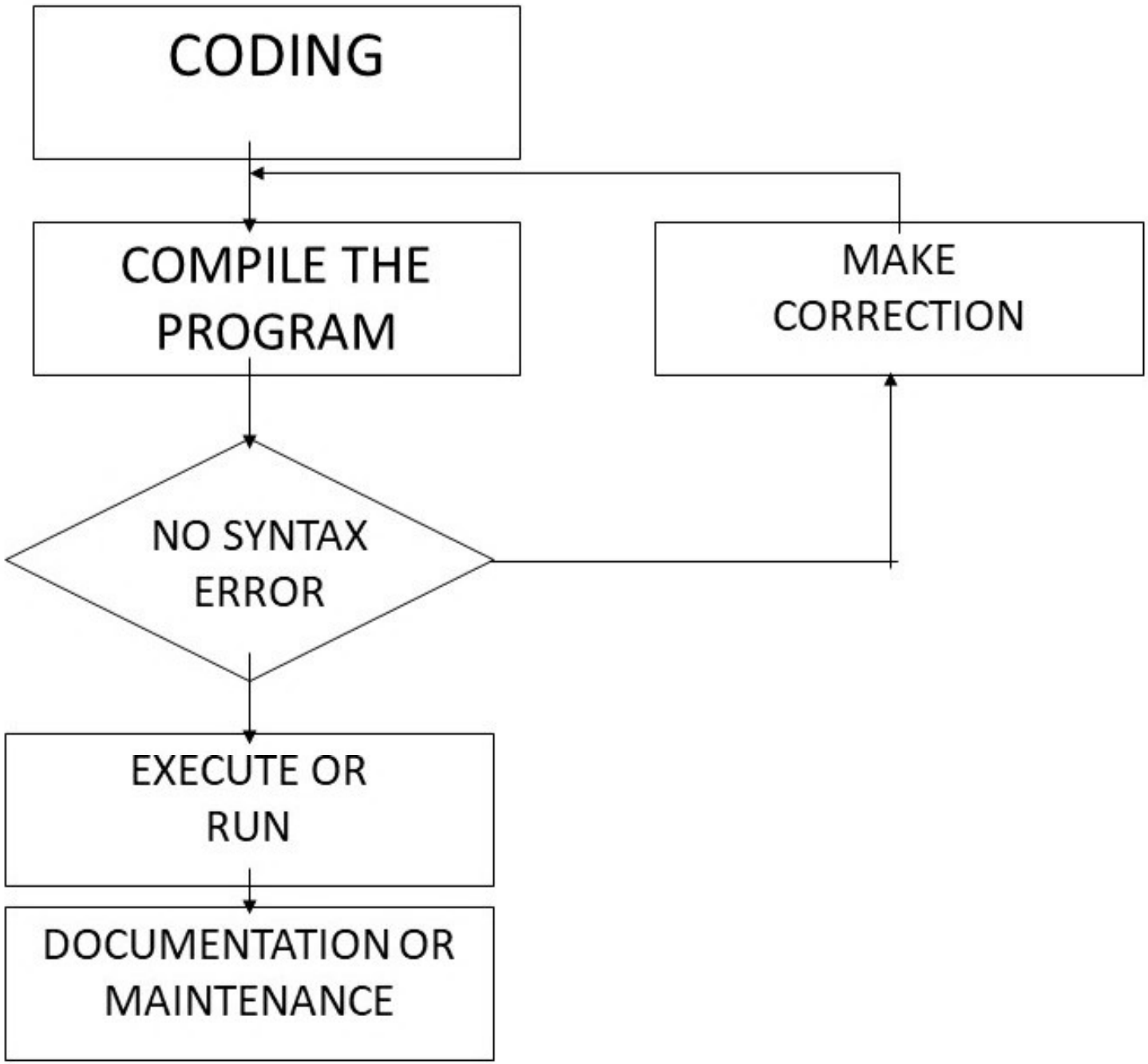
Get length, width, height
Compute volume
    volume = length * width * height
    Store volume
Display volume

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.

Programming Or Implementation Phase

- 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.


CODING
COMPILE THE PROGRAM
MAKE CORRECTION
NO SYNTAX ERROR
EXECUTE OR RUN
DOCUMENTATION OR MAINTENANCE

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

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.

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**
2. 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.
3. **Eliminate rewriting of identical process** by using modules.
4. Use techniques to improve readability including four logic structure, proper naming of variables, internal documentation and proper indentation.

Drawing Flowchart

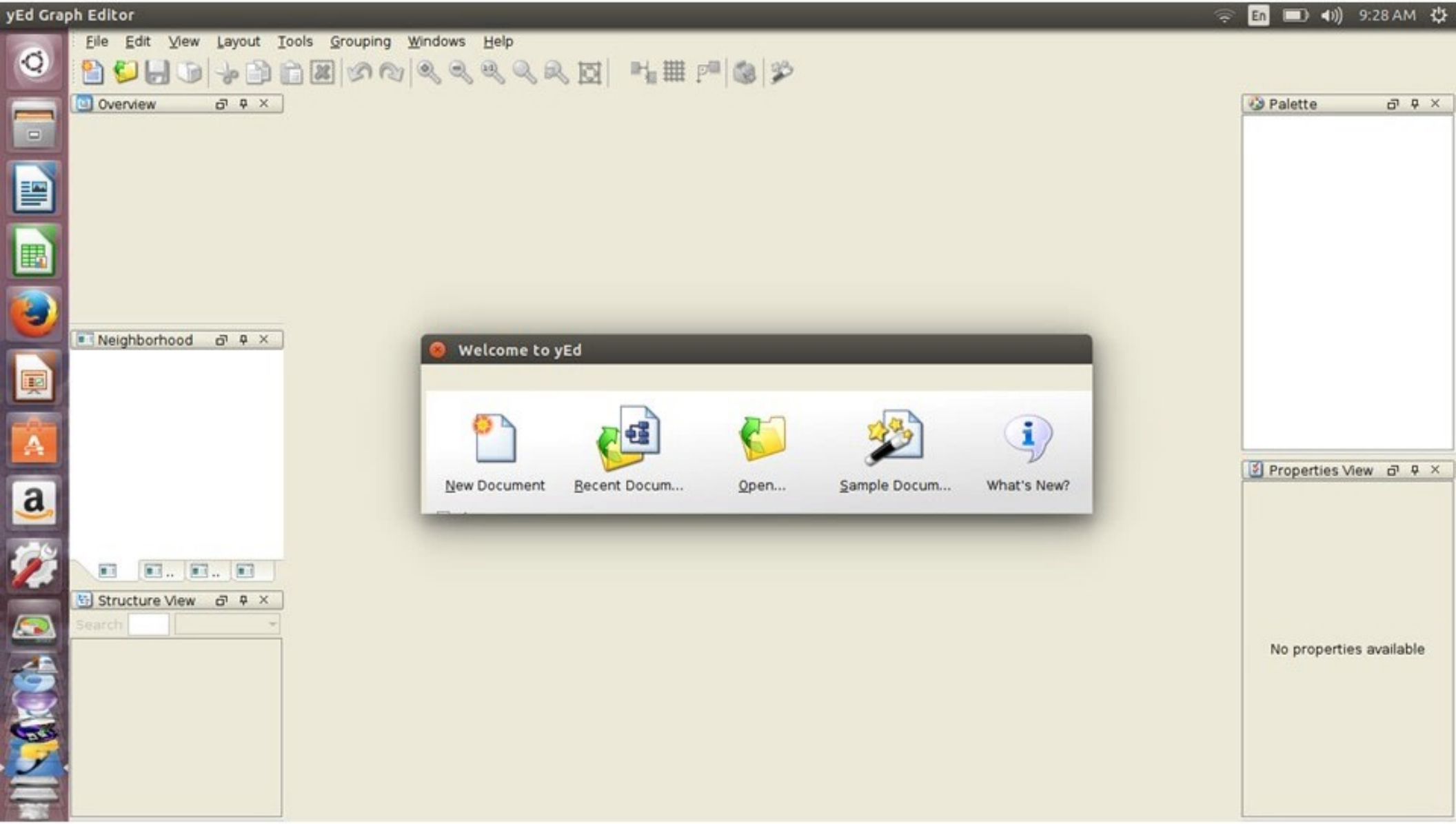
yEd – Graphical tool

yEd tool

- Used to draw flow chart

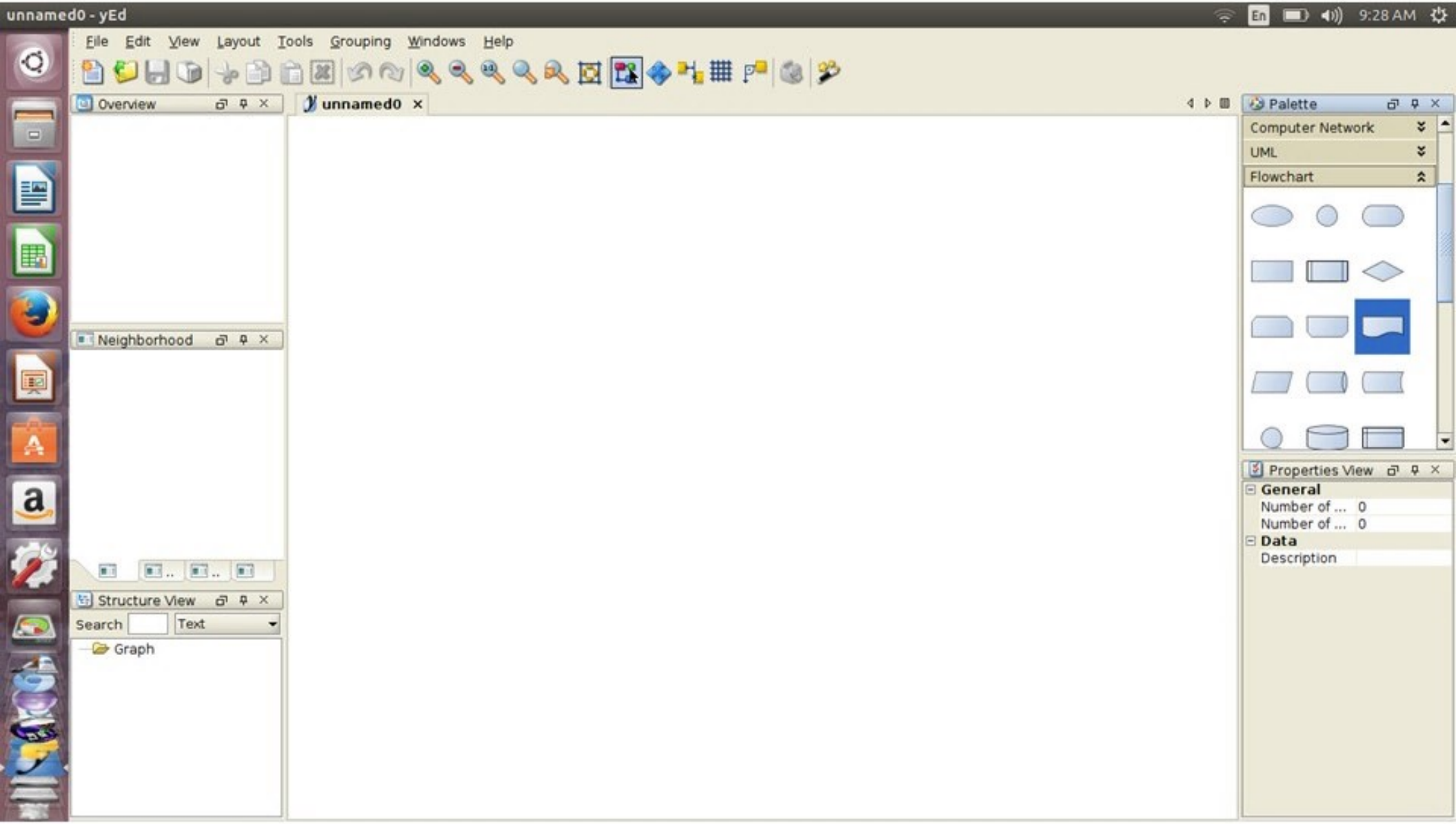
New Flow Chart

- Open Yed
- A screen as shown in next slide appears
- Choose new document in it


yEd Graph Editor
En
9:28 AM
File Edit View Layout Tools Grouping Windows Help
Overview
Palette
Neighborhood
Welcome to yEd
New Document
Recent Docum...
Open...
Sample Docum...
What's New?
Properties View
Structure View
Search
No properties available

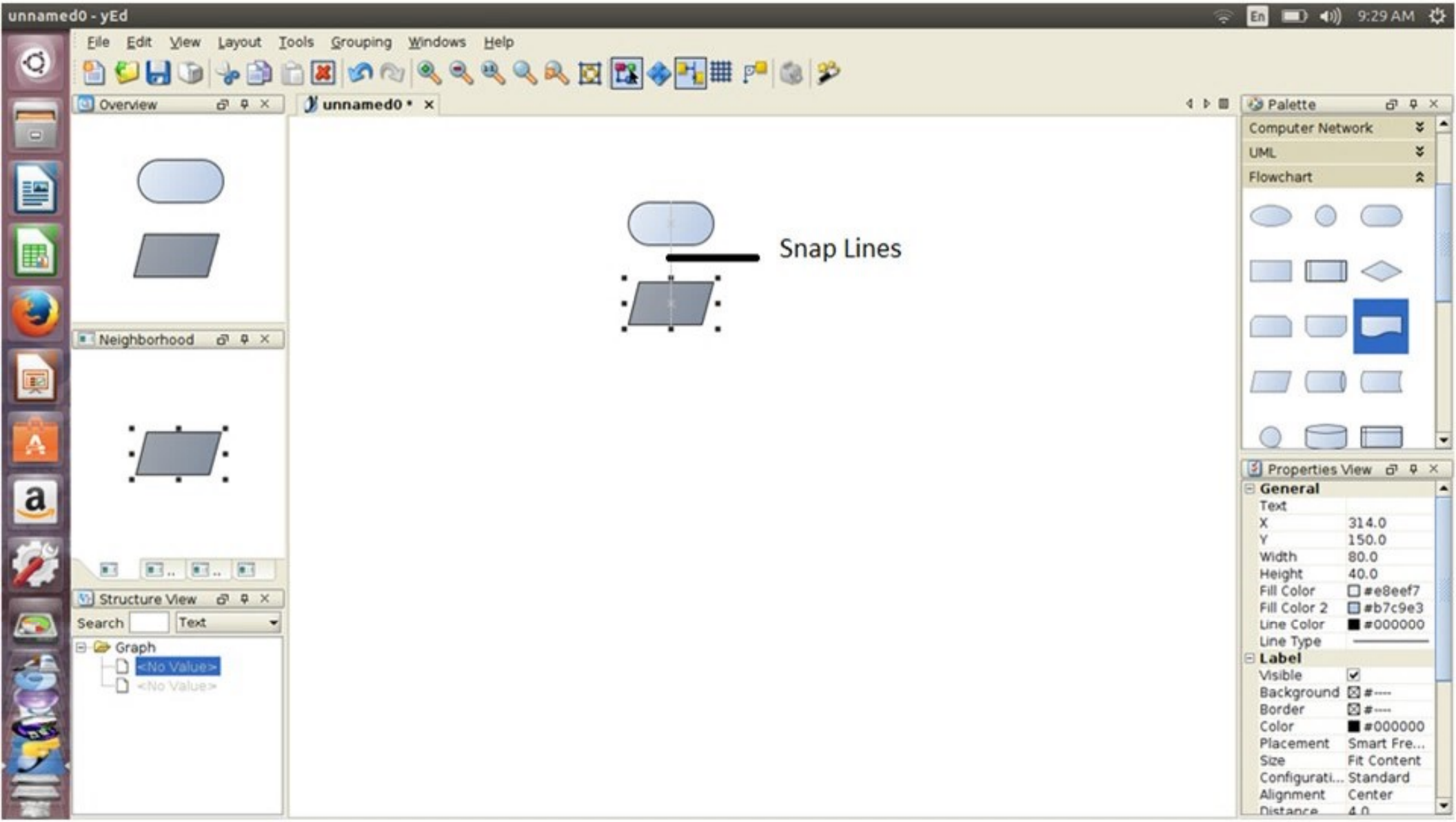
New Flow Chart

- Choose flowchart from the palette
- The tools that are used in flowchart drops down


unnamed0 - yEd
En
9:28 AM
File Edit View Layout Tools Grouping Windows Help
Overview
unnamed0
Palette
Computer Network
UML
Flowchart
Neighborhood
Properties View
General
Number of ... 0
Number of ... 0
Data
Description
Structure View
Search
Text
Graph

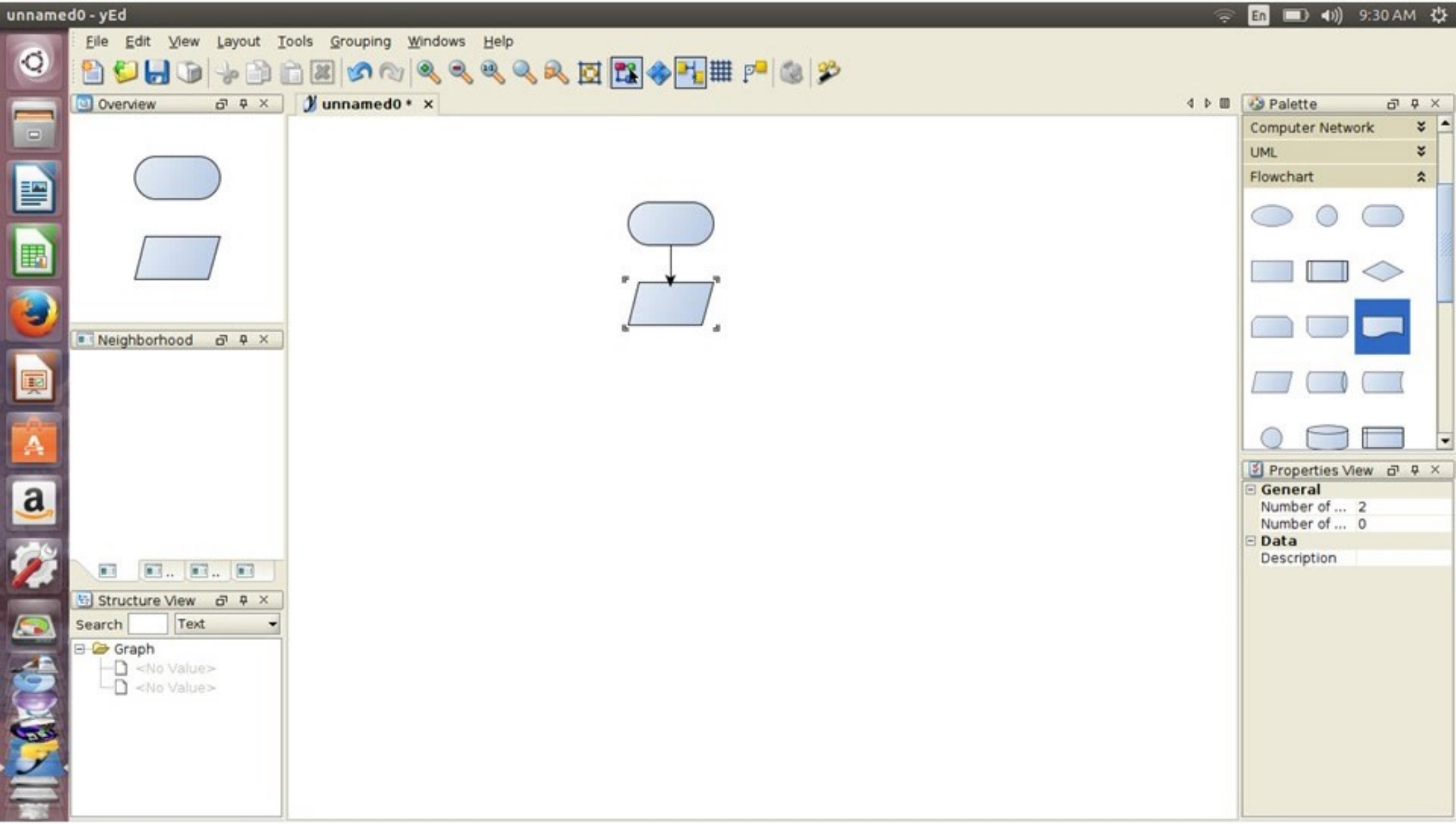
Add a new Symbol

- Drag and drop the symbol from the flowchart palette
- Snap lines help to align the new symbol with symbols already in the flowchart


unnamed0 - yEd
9:29 AM
File Edit View Layout Tools Grouping Windows Help
Overview
unnamed0 *
Palette
Computer Network
UML
Flowchart
Snap Lines
Neighborhood
Structure View
Search
Text
Graph
<No Value>
<No Value>
Properties View
General
Text
X 314.0
Y 150.0
Width 80.0
Height 40.0
Fill Color #e8eef7
Fill Color 2 #b7c9e3
Line Color #000000
Line Type
Label
Visible
Background #----
Border #----
Color #000000
Placement Smart Fre...
Size Fit Content
Configurati... Standard
Alignment Center
Distance 4.0

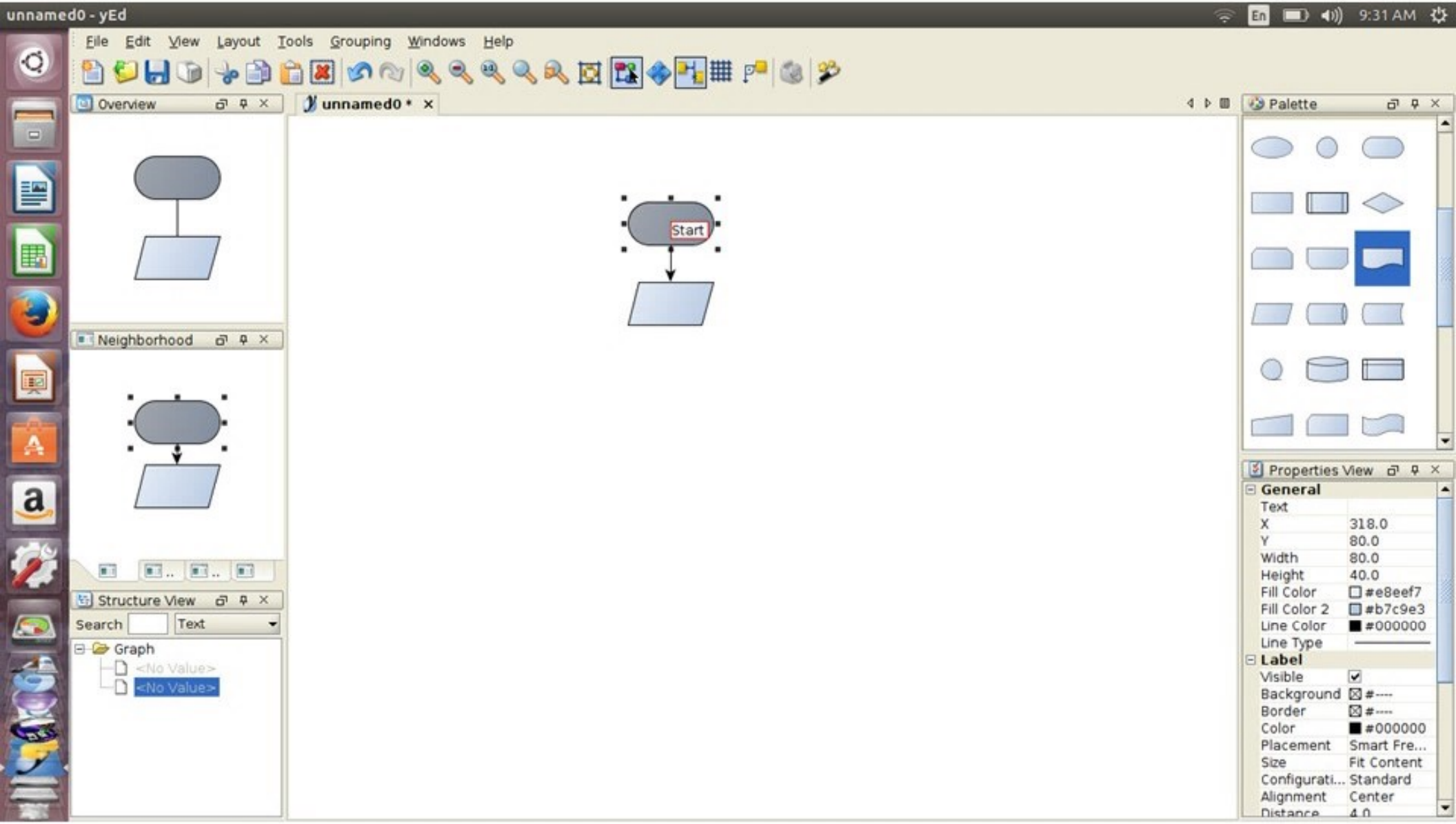
Add line to connect symbols

- To connect two symbols in the flow chart
  - Do not select any symbol
  - Place the mouse over any one of the symbol to be connected (mouse gesture will be used to draw line)
  - Drag a line to the next object that is to be connected


unnamed0 - yEd
En
9:30 AM
File Edit View Layout Tools Grouping Windows Help
Overview
unnamed0 *
Palette
Computer Network
UML
Flowchart
Neighborhood
Structure View
Search
Text
Graph
<No Value>
<No Value>
Properties View
General
Number of ... 2
Number of ... 0
Data
Description

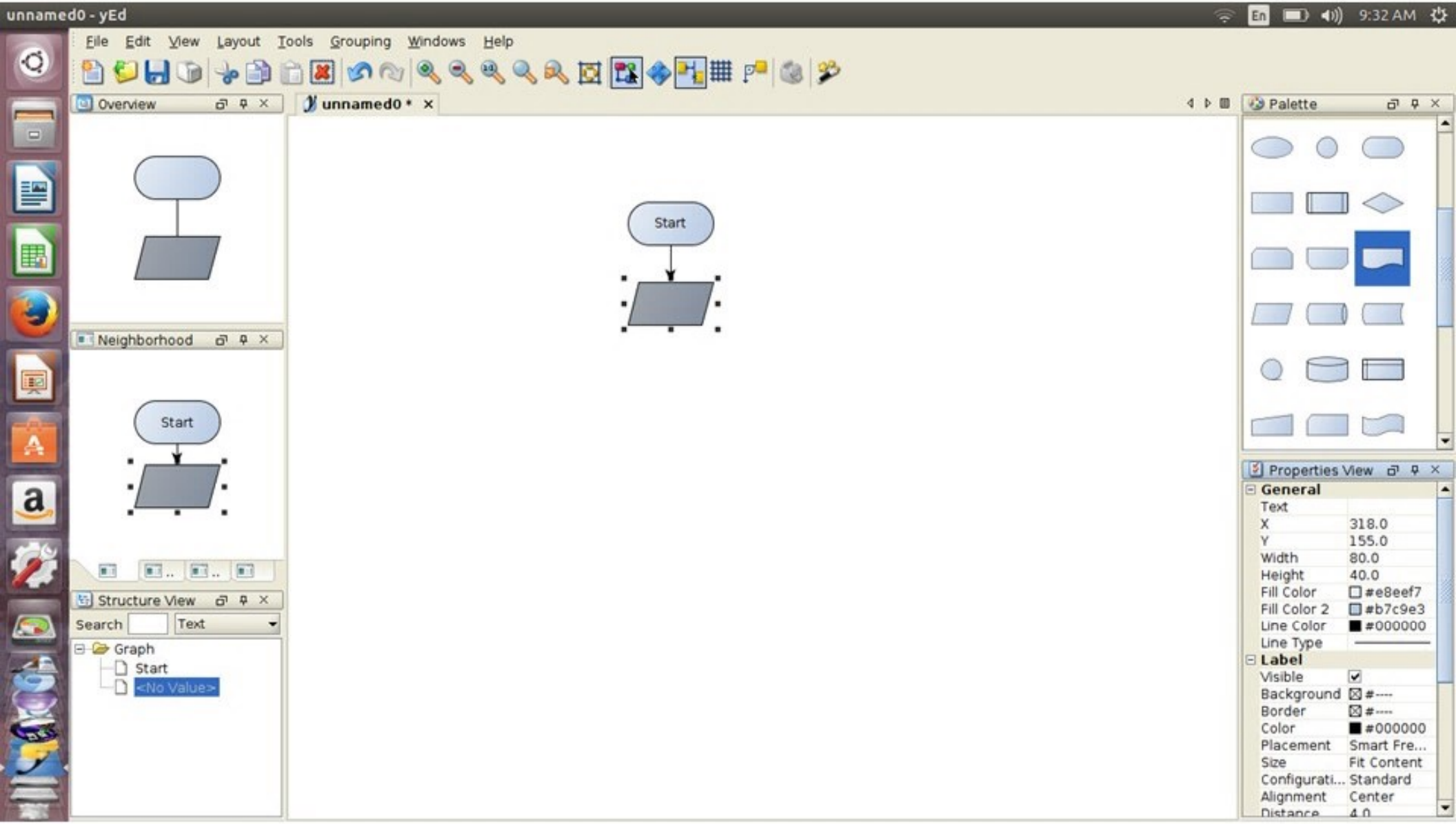
Add label to a symbol

- Select the symbol
- Press F2
- Add label inside the symbol
- To add content in more than one line press shift+Enter
- When finished
- adding content press Enter


unnamed0 - yEd
9:31 AM
File Edit View Layout Tools Grouping Windows Help
Overview
unnamed0 *
Palette
Start
Neighborhood
Structure View
Search
Text
Graph
<No Value>
<No Value>
Properties View
General
Text
X 318.0
Y 80.0
Width 80.0
Height 40.0
Fill Color #e8eef7
Fill Color 2 #b7c9e3
Line Color #000000
Line Type
Label
Visible
Background #----
Border #----
Color #000000
Placement Smart Fre...
Size Fit Content
Configurati... Standard
Alignment Center
Distance 4.0

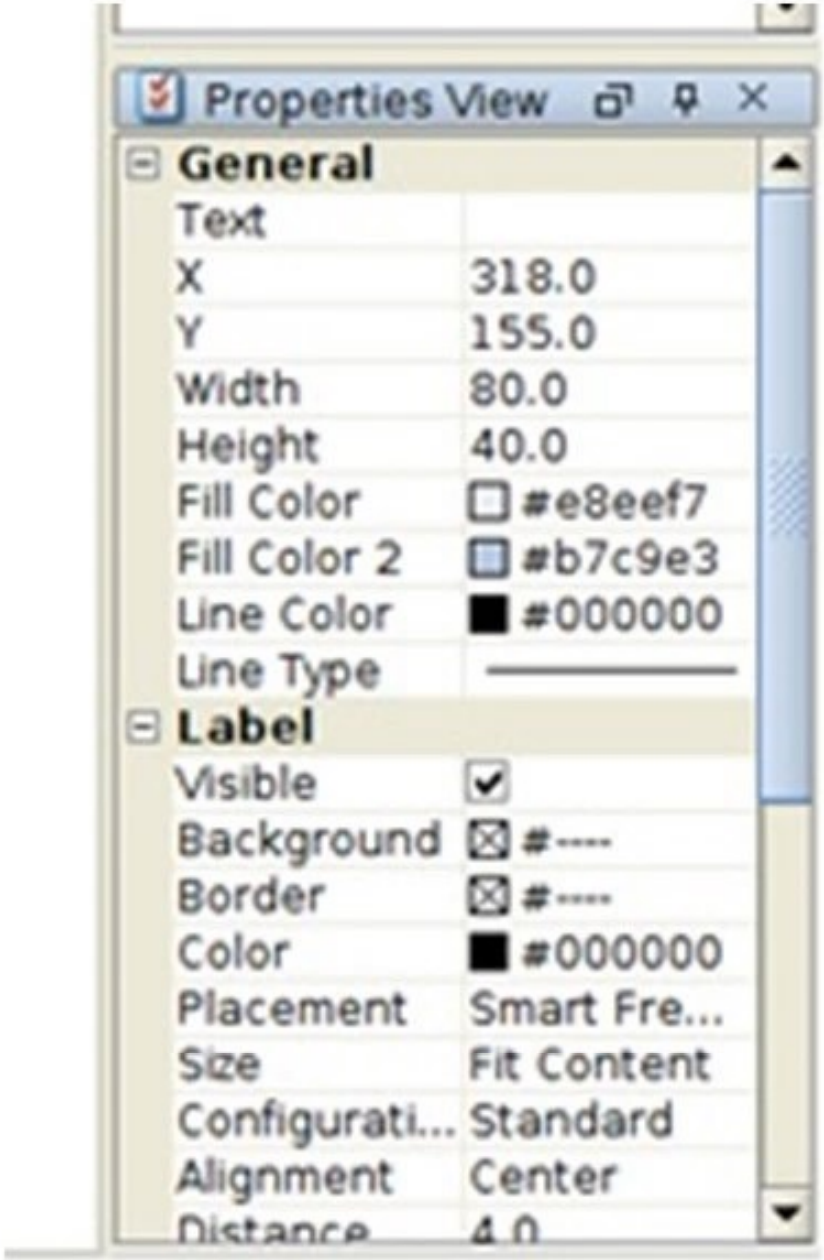
To resize an element

- Select the element
- Go to the bottom right corner to resize
- Press Shift+Ctrl keys to preserve aspect ratio while rescaling
- Snap lines help while rescaling by comparing the dimesnsion with other elements


unnamed0 - yEd
9:32 AM
File Edit View Layout Tools Grouping Windows Help
Overview
unnamed0 *
Palette
Start
Neighborhood
Start
Structure View
Search
Text
Graph
Start
<No Value>
Properties View
General
Text
X 318.0
Y 155.0
Width 80.0
Height 40.0
Fill Color #e8eef7
Fill Color 2 #b7c9e3
Line Color #000000
Line Type
Label
Visible
Background #----
Border #----
Color #000000
Placement Smart Fre...
Size Fit Content
Configurati... Standard
Alignment Center
Distance 4.0

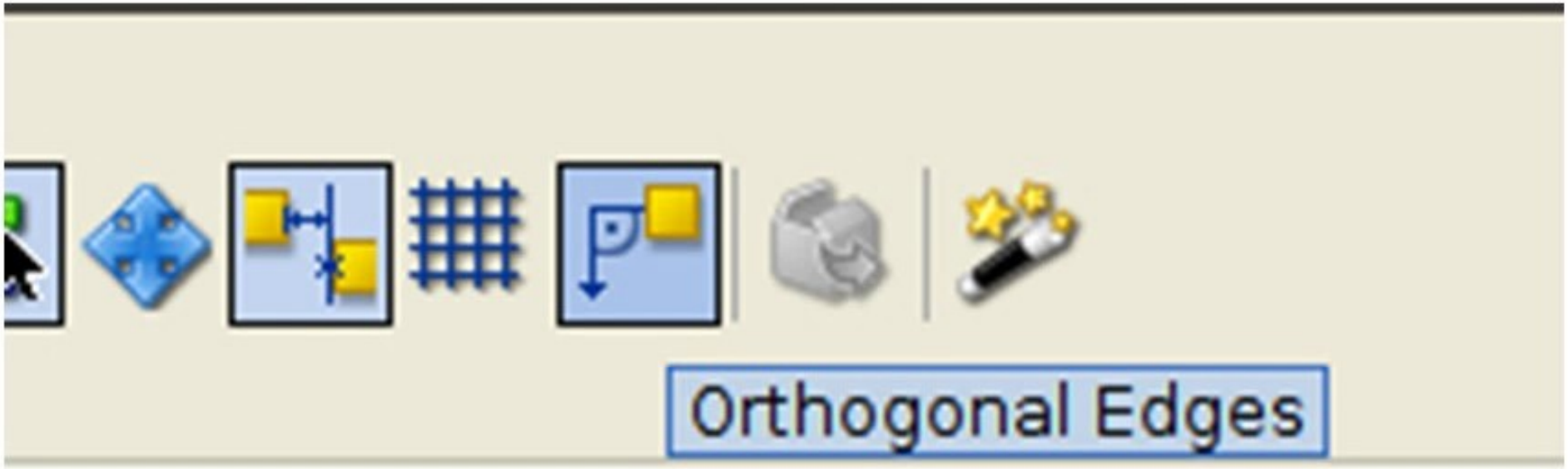
To alter properties of the elements

- Properties window can be used for this
- Background color
- Font size of labels
- Can be changed using the properties window


Properties View
General
Text
X 318.0
Y 155.0
Width 80.0
Height 40.0
Fill Color #e8eef7
Fill Color 2 #b7c9e3
Line Color #000000
Line Type
Label
Visible
Background #----
Border #----
Color #000000
Placement Smart Fre...
Size Fit Content
Configurati... Standard
Alignment Center
Distance 4.0

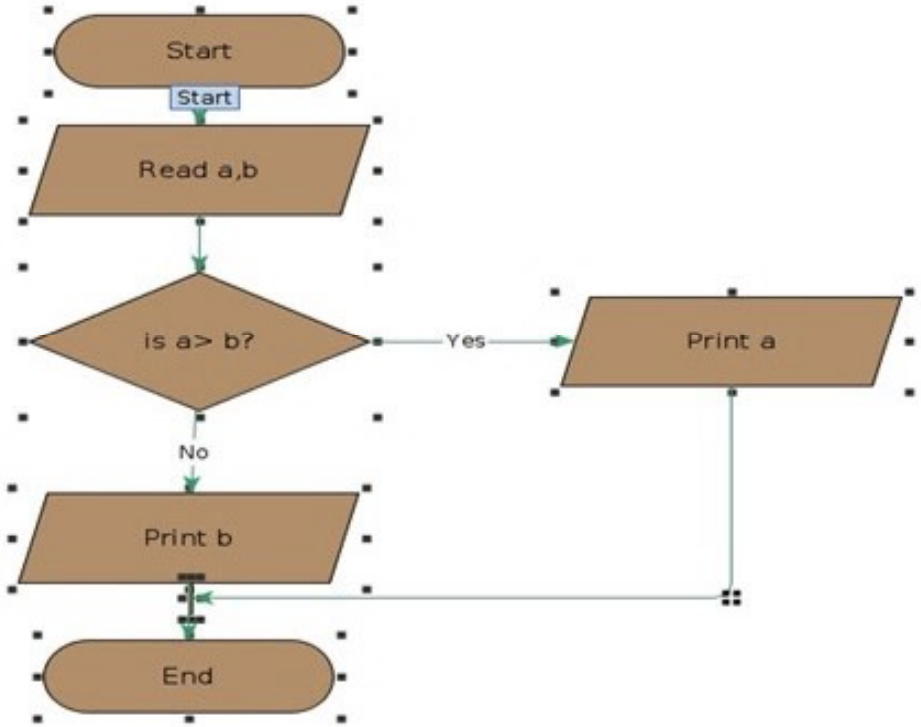
To add orthogonal edges in the flowchart

- Click on the button for orthogonal edges
- Button is at the top of the menu bar



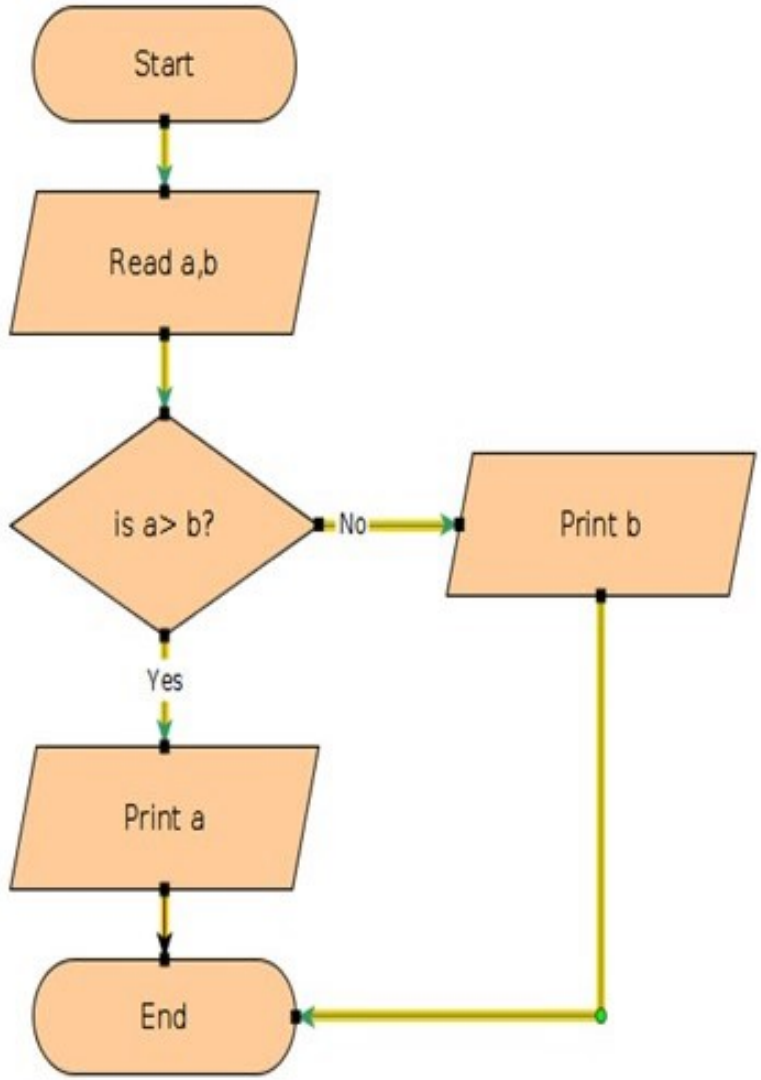
To Change properties of All Elements of Flow Chart

- Click on an element
- Press Ctrl+A
- Only Elements are chosen
- Change properties



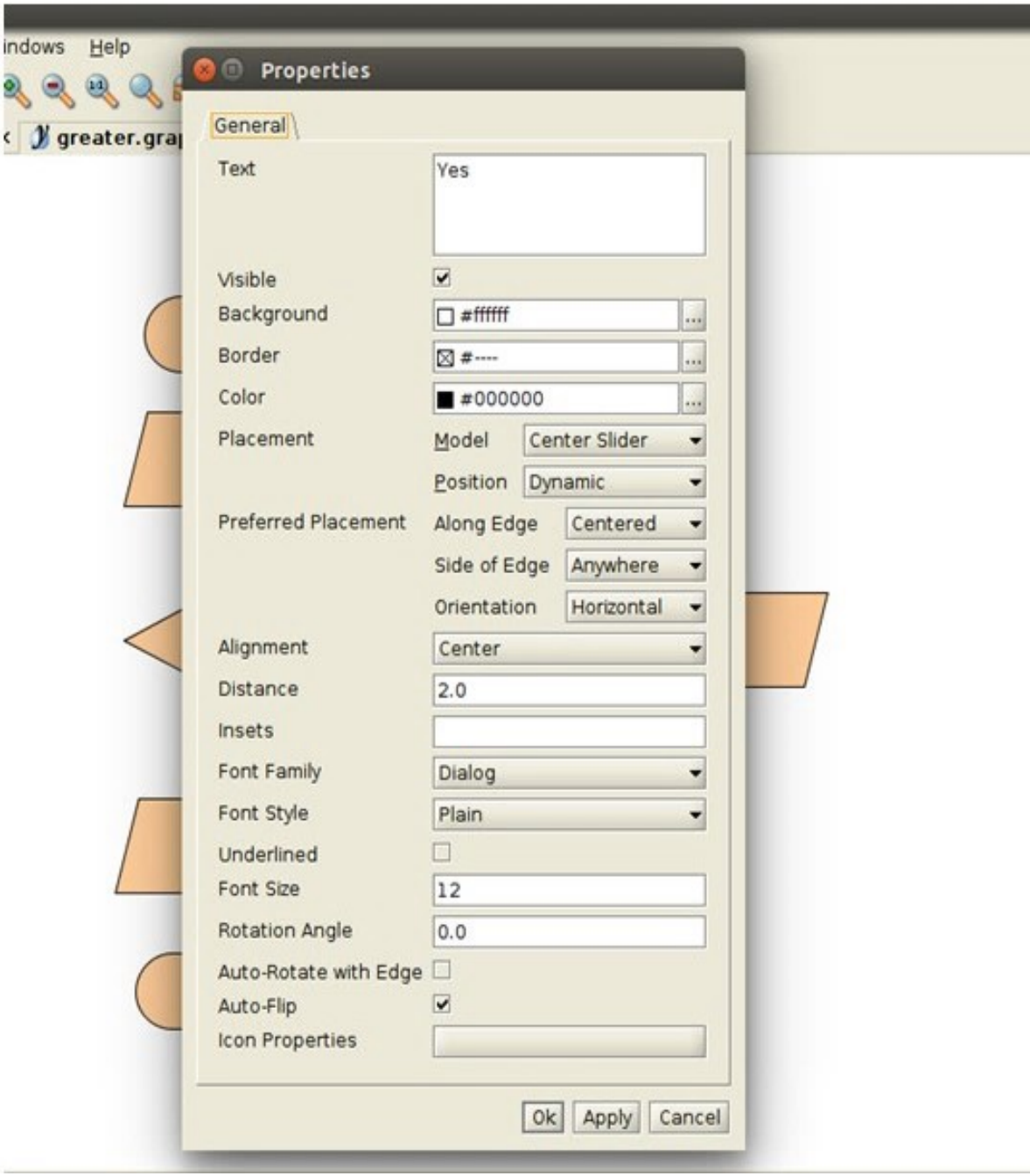
To Change properties of All lines of Flow Chart

- Click on a line
- Press Ctrl+A
- Only lines are chosen
- Change properties



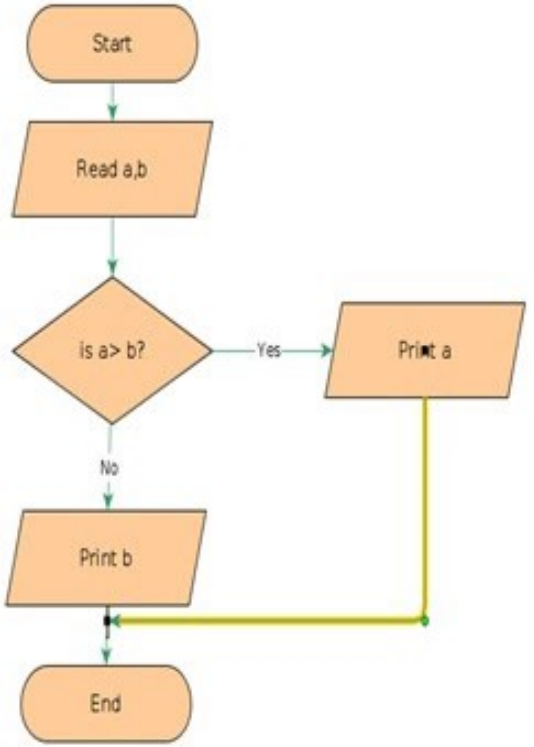
To Centre Label across the arrows

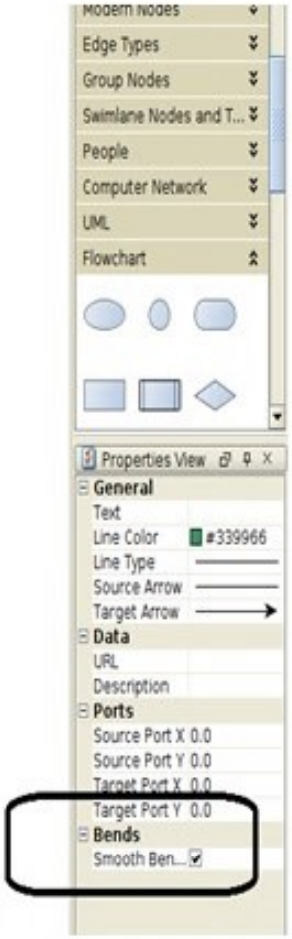
- Click on the label
- Right click and go to properties
- Select Center Slider in Model
- Set Background as White



To Smoothen the Bend in Arrows

- Click on the arrow
- Right click and go to properties
- Check the box provided for smooth bend





Flow Chart for Finding Greater of Two Numbers