GAZİ UNIVERSITY

FACULTY OF ENGINEERING DEPARTMENT OF INDUSTRIAL ENGINEERING

Lecturer : Dr Ercan Ezin

IE104-COMPUTER PROGRAMMING I

WEEK 2: Problem Solving and Algorithms

IMPORTANT ANNOUNCEMENT

- The course website is online now. Please check it out regularly.
- https://gazi-end-muh.github.io

Algorithm Denotation

1. **Plain Text**

2. **Pseudocode**

3. **Flowchart**

Example: Algorithm for Writing Your Name on a Board

1. START
2. Stand up.
3. Is your direction towards the board?
    - ➢If no, turn your direction to the board.
    - ➢If yes, go to step 4
4. Walk towards the board.
5. Did you come to the board?
    - ➢If no, go to step 4.
    - ➢If yes, go to step 6.
6. Take the chalk.
7. Write your name.
8. FINISH

Example: Algorithm for Writing Your Name on a Board

- The purpose of the example is to show the consistency and logical sequence of the steps.
- Here, orders are given by making certain inquiries and in a logical order.
- The person cannot be asked to walk without a displacement order.
- It would not be right to give an order to write his/her name without the chalk.
- It has also been checked whether the operation has done or not with asking the right questions.

Example: Algorithm for Writing Your Name on a Board

- Actually, the computer cannot perform such tasks.
- The users give some inputs to computers.
- And the computers produce an output by performing mathematical and logical operations on the inputs according to the steps given by the programmer.

Pseudocode Notation

- It is a language consisting of simple commands that can be easily translated into a programming language and everyone can understand.
- The main function of pseudocode is to *create* and *discuss* the *algorithm* before proceeding the development of a program.
- Pseudocodes are written with the terms such as *if, like, and expressions* (> = <) in a direct speech and under programming logic.
- Appropriate names or variables are selected to represent the elements to be used in the program.
- Arithmetic operations are performed using algebraic notation and decision.

Example: Algorithm for Sum of Two Numbers

1. START
2. Enter first number.
3. Enter second number.
4. Sum the two numbers.
5. Print the sum value of numbers.
6. FINISH

How can we convert this line to a Pseudo Code Notation?

Example: Algorithm for Sum of Two Numbers

PLAIN TEXT

1. START
2. Enter first number.
3. Enter second number.
4. Sum the two numbers.
5. Write the sum value of numbers.
6. FINISH

PSEUDOCODE

```
1. START
2. Read X.
3. Read Y.
4. $T = X + Y$.
5. Write the value of T.
6. FINISH
```

Example: Algorithm Calculating Area of a Triangle

1. START
2. Enter the base value.
3. Enter the height value.
4. Multiply the base by the height and divide the result by two.
5. Write the result
6. FINISH

How can we convert it to a Pseudo Code Notation?

Example: Algorithm Calculating Area of a Triangle

PLAIN TEXT

1. START

2. Enter the base value.

3. Enter the height value.

4. Multiply the base by the height and divide the result by two.

5. Write the result

6. FINISH

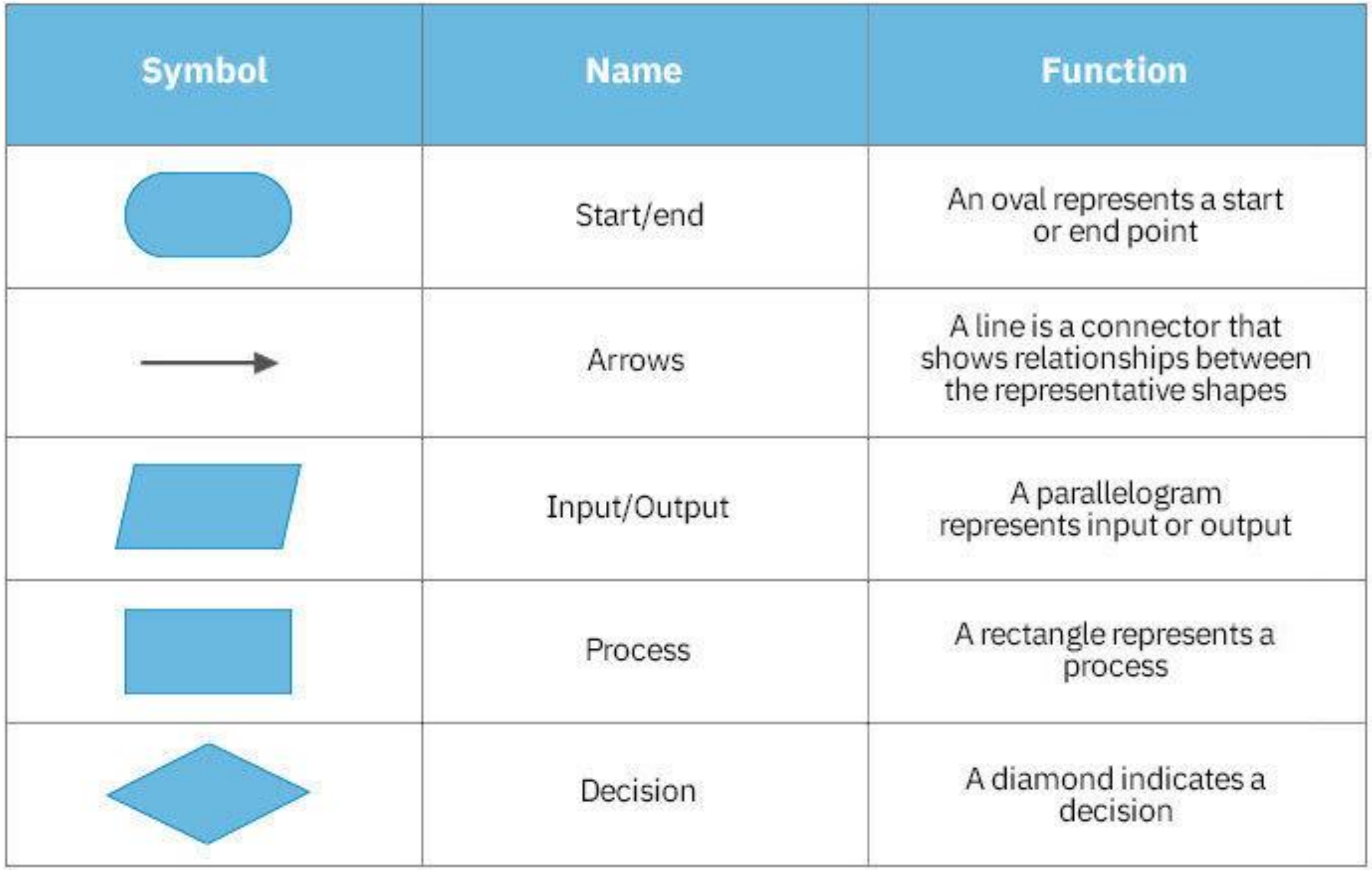
PSEUDOCODE

```
1. START
2.Read b.
3.Read h.
4. $A = (b*h)/2$.
5. Write the value of A.
6. FINISH
```

Flowcharts

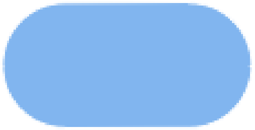
- An algorithm expressed in visual shapes and symbols is called "Flowcharts".
- Flowchart symbols are determined by ANSI (American National Standards Institute) and are used all over the world as a standart notation.
- Since the algorithm is written in a natural language, it may not be understood by everyone or might be interpreted incorrectly.
- However, with flowcharts, each figure has a standard meaning in flow charts. Therefore, it is expected to be interpreted the same by everyone.

Flowchart Symbols



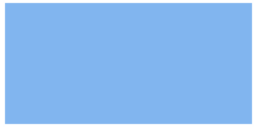
Symbol	Name	Function
⬭	Start/end	An oval represents a start or end point
→	Arrows	A line is a connector that shows relationships between the representative shapes
▱	Input/Output	A parallelogram represents input or output
▭	Process	A rectangle represents a process
◆	Decision	A diamond indicates a decision

There are many flowchart symbols available, keep this for your future references and use it if the basic ones are not sufficient.



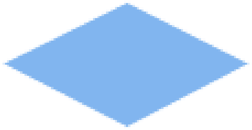
Terminator

Indicates the beginning or end of a program flow in your diagram.



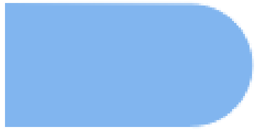
Process

Indicates any processing function.



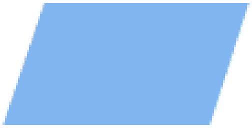
Decision

Indicates a decision point between two or more paths in a flowchart.



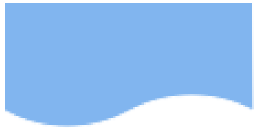
Delay

Indicates a delay in the process.



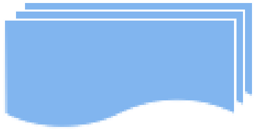
Data

Can represents any type of data in a flowchart.



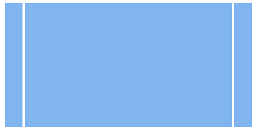
Document

Indicates data that can be read by people, such as printed output.



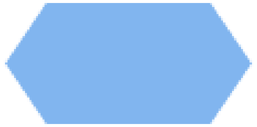
Multiple documents

Indicates multiple documents.



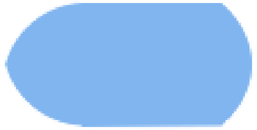
Subroutine

Indicates a predefined (named) process, such as a subroutine or a module.



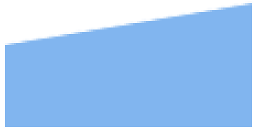
Preparation

Indicates a modification to a process, such as setting a switch or initializing a routine.



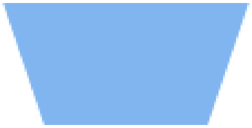
Display

Indicates data that is displayed for people to read, such as data on a monitor or projector screen.



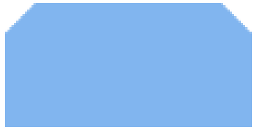
Manual input

Indicates any operation that is performed manually (by a person).



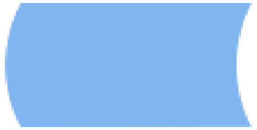
Manual loop

Indicates a sequence of commands that will continue to repeat until stopped manually.



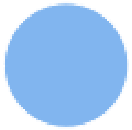
Loop limit

Indicates the start of a loop. Flip the shape vertically to indicate the end of a loop.



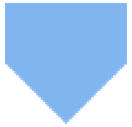
Stored data

Indicates any type of stored data.



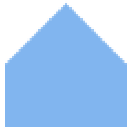
Connector

Indicates an inspection point.

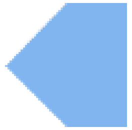


Off-page connector

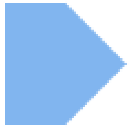
Use this shape to create a cross-reference and hyperlink from a process on one page to a process on another page.



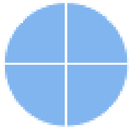
Off-page connector



Off-page connector

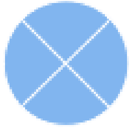


Off-page connector



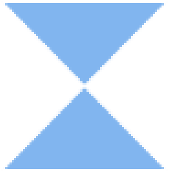
Or

Logical OR



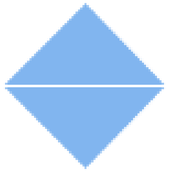
Summing junction

Logical AND



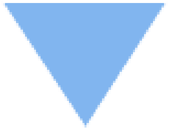
Collate

Indicates a step that organizes data into a standard format.



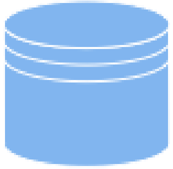
Sort

Indicates a step that organizes items list sequentially.



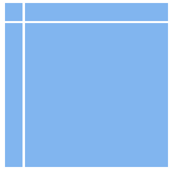
Merge

Indicates a step that combines multiple sets into one.



Database

Indicates a list of information with a standard structure that allows for searching and sorting.



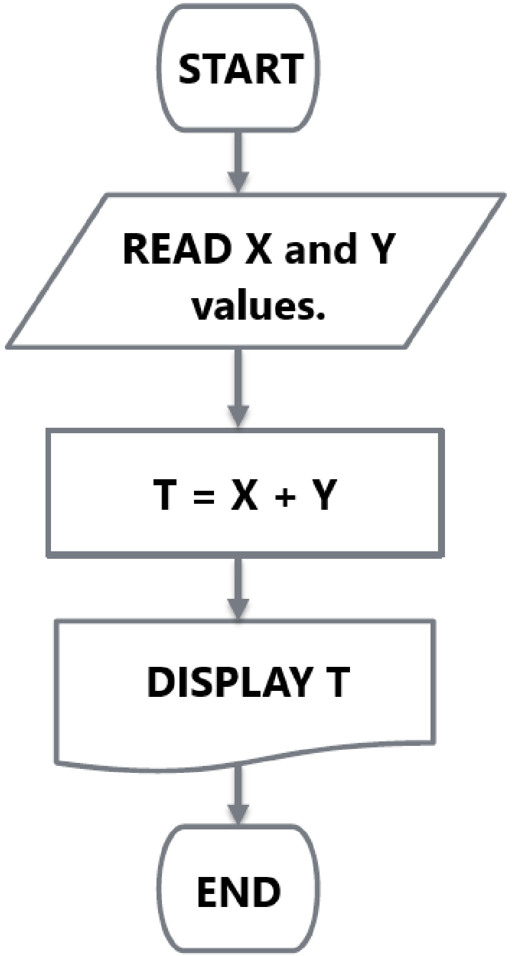
Internal storage

Indicates an internal storage device.

Example: Flowchart for Sum of Two Numbers

PSEUDOCODE

```
1. START
2. Read X.
3. Read Y.
4. T = X + Y.
5. Display the value of T.
6. FINISH
```



START

READ X and Y values.

T = X + Y

DISPLAY T

END

Example: Flowchart for Calculating Triangle Area

PSEUDOCODE

```
1. START
2. Read b.
3. Read h.
4. $A = (b * h )/ 2$
5. Print the value of A.
6. FINISH
```

BUILD FLOWCHART YOURSELF

?

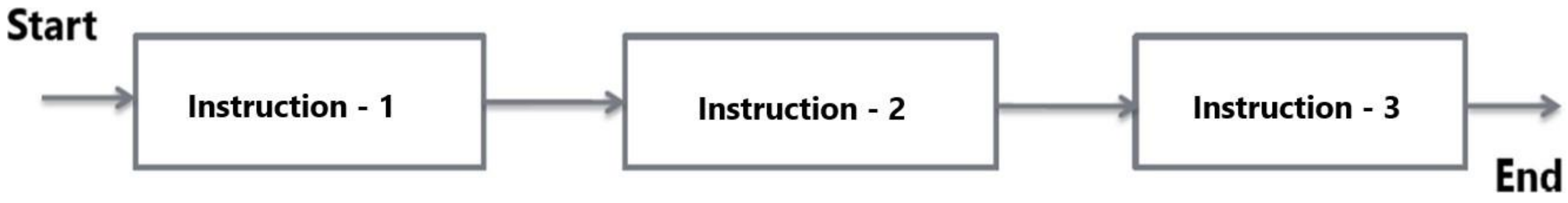
Logical Structures

Regardless of the programming language used in the development of a computer program, three simple logical structures are generally used in the flowcharts of these programs.

1. Sequential Structures
2. Decision Making Structures
3. Repetitive Structures

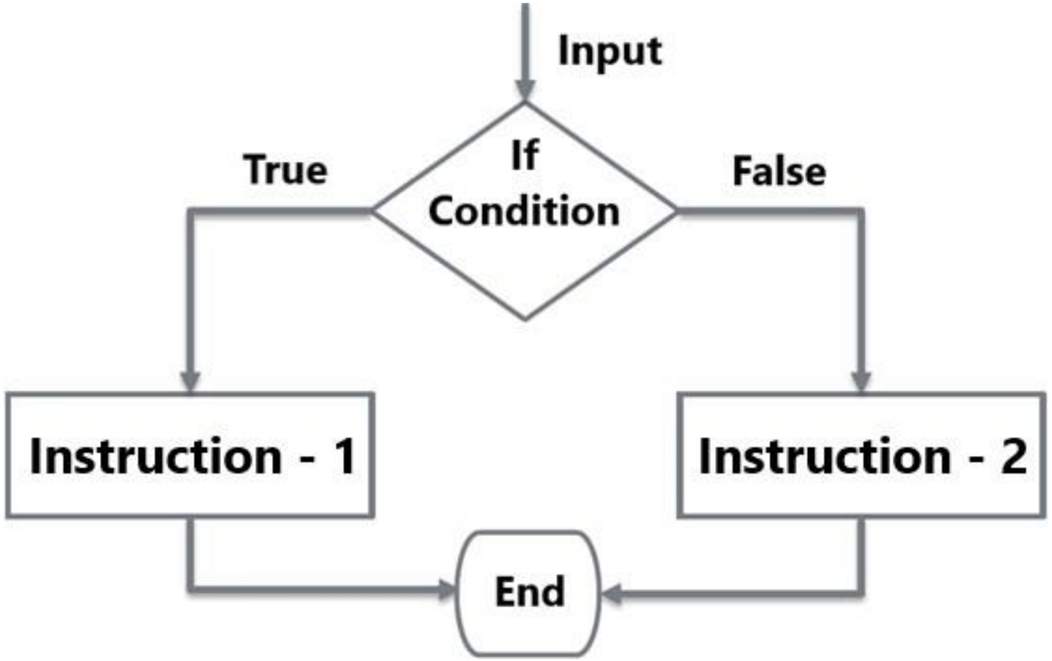
Sequential Structures

- The sequential structure emphasizes that every operation in the program should be prepared and performed in logical order. A second operation cannot start until this structure is finished.



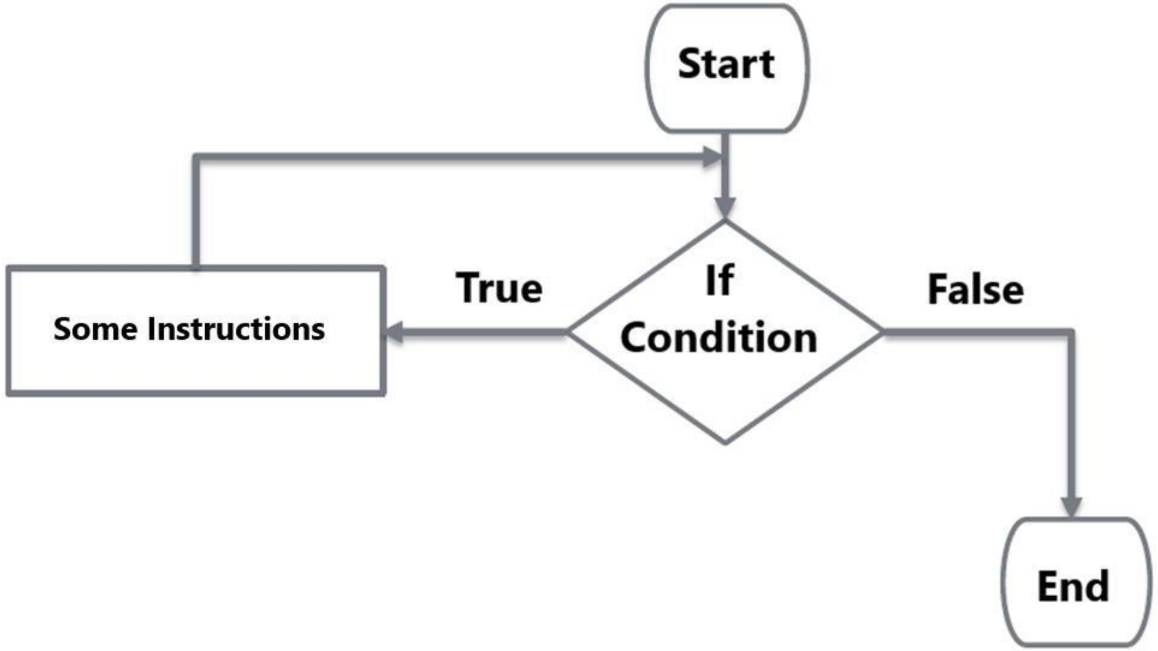
Decision Making Structures

- In modules with more than one sequential structure option, DM structure determines which sequential structure will be selected under which conditions.



Repetitive Structures

- Within the algorithm, if some rows are processed repeatedly, there exists a loop. Loops are referenced to define actions that continue as long as a certain condition is valid.



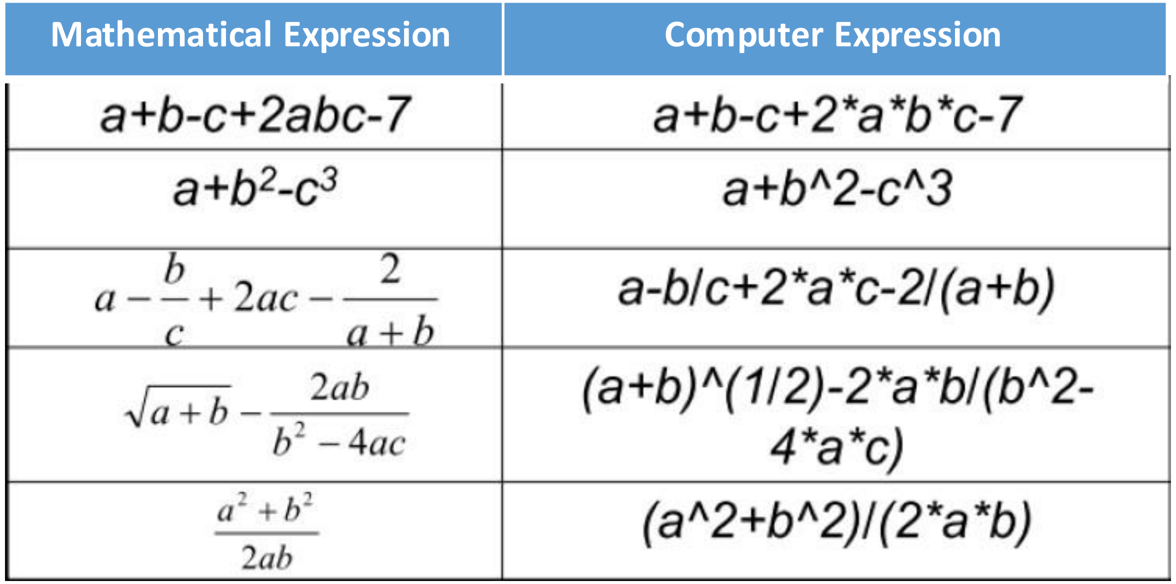
Operations and Operators

Operations are divided into 3:

1. Mathematical Operations
   - Basic Arithmetic Operations: Addition, subtraction, multiplication, division.
   - Mathematical Functions: Exponential, logarithmic, trigonometric, hyperbolic etc.
2. Comparison Operations
3. Logical Operations

Mathematical Operations

Operations	Notation
Sum	a+b
Subtraction	a-b
Multiplication	a*b
Division	a/b
Exponention	a^b



Mathematical Expression	Computer Expression
$a+b-c+2abc-7$	a+b-c+2*a*b*c-7
$a+b^2-c^3$	a+b^2-c^3
$a-\frac{b}{c}+2ac-\frac{2}{a+b}$	a-b/c+2*a*c-2/(a+b)
$\sqrt{a+b}-\frac{2ab}{b^2-4ac}$	(a+b)^(1/2)-2*a*b/(b^2-4*a*c)
$\frac{a^2+b^2}{2ab}$	(a^2+b^2)/(2*a*b)

Comparison Operations

- These are the operations that control whether the variables are greater than, less than, and equal.

Operation Symbol	Meaning
=	Equals
<>	Not equal
>	Bigger than
<	Less than
>=	Equal or Bigger
<=	Equal or Less

Logical Operations

- The "And, Or, Not" operators are used both in mathematical operations and conditional statements.

Logical Operation	Symbol
And	&&
Or	||
Not	!

- The conjunction AND means that both conditions are met. If at least one of the propositions associated with AND is false, the result will be false.
- If one of the conditions associated with OR is true, the result will be true.
- The NOT conjunction makes true condition false and false condition true.

Logical Operations

- Example: Among the male staff working in the software department, those over the age of 30 will be displayed on the screen.

- If $\underbrace{\text{(perGender=Male)}}_{\text{1. CONDITION}}$ AND $\underbrace{\text{(perAge>30)}}_{\text{2. CONDITION}}$ than display on screen.

Some Terms Used in the Algorithm Literature

1. Identifier
2. Variable
3. Assignment
4. Counter
5. Loops

Identifier

- It is created by the programmer.
- Words that are used to name variables, constants, registers, custom information types, etc. in the program.
- Identifiers should contain keywords of the attribute they represent.
- In C# language, 26 letters from the English alphabet A-Z or a-z and numbers 0-9 can be used.
- Only Underscore(_) symbols can be used.
- Identifier names can only begin with a letter or an underscore.
- Identifier name cannot begin with a number or contain only numbers.

Variable

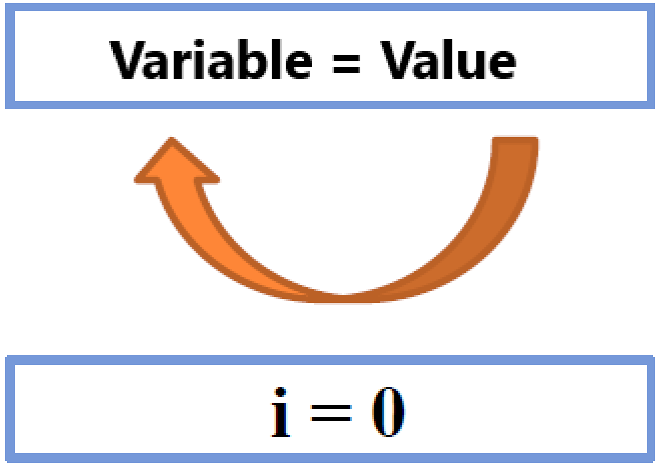
- The information/memory fields that take different values each time every program run.
- Variable naming should be done in accordance with descriptive rules.

Examples:

- Variable which the *long edge of the rectangle* is represented as:
  - ➢long_edge,
  - ➢LongEdge,
  - ➢longedge
- The variable which a student's name is represented:
  - ➢name,
  - ➢student_name,
  - ➢studentName

Assignments

- It is the process of assigning values to variables. Values assigned to variables can be reused later.



Variable = Value

$i = 0$

The value on the right is transferred to the variable. If there is a previous value of the variable, it will be deleted.

Counters

- Some operations may need to be performed a certain number of times and the produced values may need to be counted.
- Such counting operations are performed with "Counters" in the algorithm.
- Counters are variables.

Counter = Counter + 1

In this operation, 1 is added to the Counter variable and the result is transferred to itself.

Loops

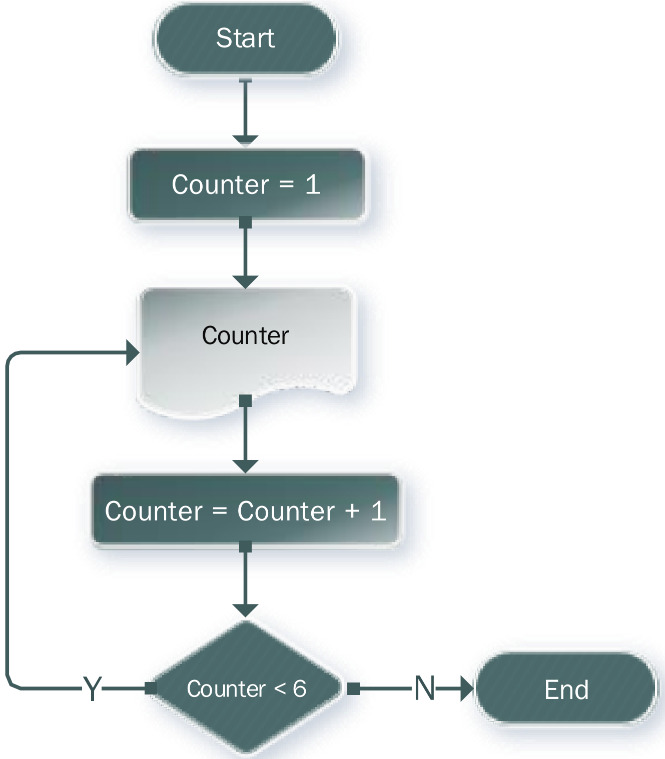
- In many programs, some operations are performed *with certain sequential values* or performed *a certain number of times*.
- Process flow cycles that perform certain transaction blocks in a program with the given number of times are called *"Loops".*
- For example, in the program that calculates the sum of odd numbers between 1 and 1000, instead of T=1+3+5..., a loop increasing by 2 is established between 1-1000, and the loop variable is added sequentially.

Example – 1

- Write the pseudocode and flowchart of a program that displays the number between 1-5.

PSEUDOCODE

```
1. START
2. Counter = 1
3. Display Counter.
4. Counter = Counter + 1
5. If Counter<6 go to step 3.
6. END
```



Example – 1

- Write the pseudocode and flowchart of a program that displays the number between 1-5.

PSEUDOCODE

```
1. START
2. Counter = 1
3. Display Counter.
4. Counter = Counter + 1
5. If Counter<6 go to step 3.
6. END
```

Variable Watch List

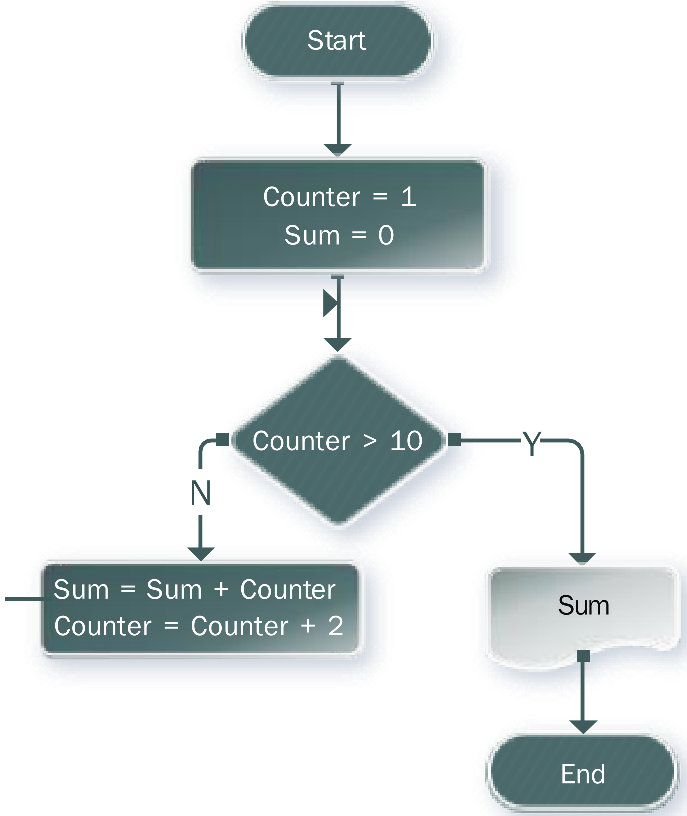
Old Counter	New Counter	Screen
1	2	1
2	3	2
3	4	3
4	5	4
5	6	5

Example – 2

- Write the pseudocode and flowchart of a program that finds the sum of odd numbers between 0-10 and prints the sum value.

PSEUDOCODE

```
1. START
2. Counter = 1, Sum=0
3. If Counter>10 goto step 7.
4. Sum = Sum + Counter
5. Counter = Counter + 2.
6. Goto 3.
7. Print Sum.
8. END
```



Example – 2

- Write the pseudocode and flowchart of a program that finds the sum of odd numbers between 0-10.

PSEUDOCODE

```
1. START
2. Counter = 1, Sum=0
3. If Counter>10 goto step 7.
4. Sum = Sum + Counter
5. Counter = Counter + 2.
6. Goto 3.
7. Print Sum.
8. END
```

Variable Watch List

Old Counter	Old Sum	New Counter	New Sum
1	0	3	1
3	1	5	4
5	4	7	9
7	9	9	16
9	16	11	25
11			

THE END

CHECK OUT COURSE WEBSITE: HTTPS://GAZI-END-MUH.GITHUB.IO

GOT ANY QUESTIONS LATER?

SEND ME AN EMAIL: DR.ERCAN.EZIN@GMAIL.COM