



**Faculty of Computer and Artificial Intelligence  
Sadat University**



# **Analysis and Design of Algorithms (CS 302)**

## **Lecture :1**

---

### **“Introduction”**

**Dr.Sara A Shehab**

# Introduction

---

## ❑ What is an algorithm?

A simple, unambiguous, mechanical procedure to carry out some task.

## ❑ Why algorithm instead of program?

1. Writing an algorithm is simpler (we don't need to worry about the detailed implementation, or the language syntax).
2. An algorithm is easier to read than a program written in, for instance, C.

# Introduction (Cont..)

---

## □ How to represent an algorithm?

1. Give a description in your own language, e.g. English, Spanish, ...
2. Pseudo code
3. Graphical

# Algorithm Efficiency

---

- Some algorithms are more efficient than others
- The complexity of an algorithm is a function describing the efficiency of the algorithm in terms of the amount of data the algorithm must process.
- There are two main complexity measures of the efficiency of an algorithm

**1. Time Complexity**

**2. Space Complexity**

# Time Complexity

---

- Time Complexity is a function describing the amount of time an algorithm takes in terms of the amount of input to the algorithm
- "Time" can mean the number of memory accesses performed, the number of comparisons between integers, the number of times some inner loop is executed.

# Space Complexity

---

- Space complexity is a function describing the amount of memory (space) an algorithm takes in terms of the amount of input to the algorithm.
- Space complexity is sometimes ignored because the space used is minimal and/or obvious, but sometimes it becomes as important an issue as time.

# Big Oh notation

---

- For run time complexity analysis we use big Oh notation.
- We have chosen to use big Oh notation for a few reasons, the most important of which is that it provides an abstract measurement by which we can judge the performance of algorithms without using mathematical proofs.

## ❑ The following list explains some of the most common big Oh notations:

- $O(1)$  constant: the operation doesn't depend on the size of its input, such as adding a node to the tail of a linked list.
- $O(n)$  linear: the run time complexity is proportionate to the size of  $n$ .
- $O(\log n)$  logarithmic: normally associated with algorithms that break the problem into smaller chunks per each invocation, such as searching a binary search tree.
- $O(n \log n)$  just  $n \log n$ : usually associated with an algorithm that breaks the problem into smaller chunks per each invocation, and then takes the results of these smaller chunks and stitches them back together, such as quick sort.



# Pseudocode

---

- Pseudocode is not an actual programming language.
- So it cannot be compiled into an executable program.
- It uses short terms or simple English language syntaxes to write code for programs before it is actually converted into a specific programming language.
- And there is no pseudocode standard syntax and so at times it becomes slightly confusing when writing Pseudocode and so let us understand pseudo code with an example.

# Pseudocode Example

## Pseudocode Example

Design the algorithm and flowchart that finds and display the larger of the two numbers given different from each other.

```
1
2 BEGIN
3
4 Number n1,n2
5
6 Input n1
7
8 Input n2
9
10 IF (n1>n2) THEN
11 OUTPUT(n1+" is higher")
12 ELSE IF(n2>n1)
13 OUTPUT(n2+" is higher")
14 ELSE
15 OUTPUT("n1=n2")
16 END IF
17 END
18
```

# Pseudocode Example

## Pseudocode Example

Perform the application that calculates the area of the triangle whose height and base length entered by the keyboard.

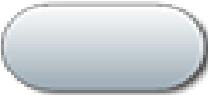




```
1
2 BEGIN
3
4 Number b,h
5 ,area
6 Input b
7
8
9 Input h
10
11 area=(b*h)/2
12 OUTPUT (Area of Triangle is "+area)
13 END
14
15
16
17
18
19
20
21
22
23
```

# Flow-Chart

---

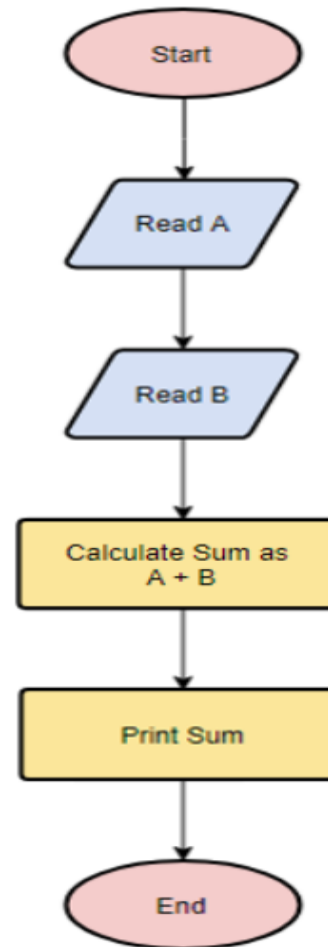
A flowchart is simply a graphical representation of steps. It shows steps in sequential order and is widely used in presenting the flow of algorithms, workflow or processes. Typically, a flowchart shows the steps as boxes of various kinds, and their order by connecting them with arrows.

# Flow-Chart

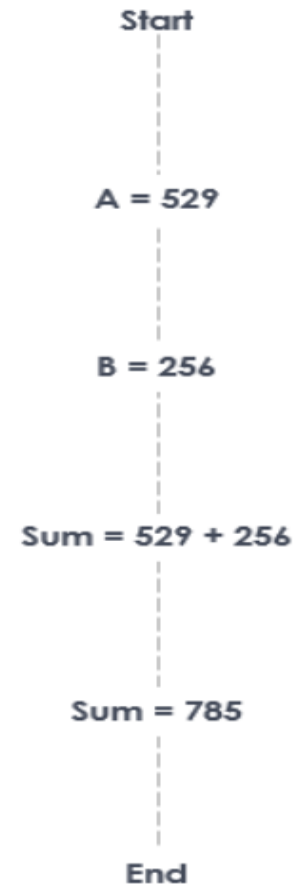
Symbol	Name	Function
	Start/end	An oval represents a start or end point.
	Arrows	A line is a connector that shows relationships between the representative shapes.
	Input/Output	A parallelogram represents input or output.
	Process	A rectangle represents a process.
	Decision	A diamond indicates a decision.

# Flow-Chart Example

Here is an example that shows how flowchart can be used in showing a simple summation process.



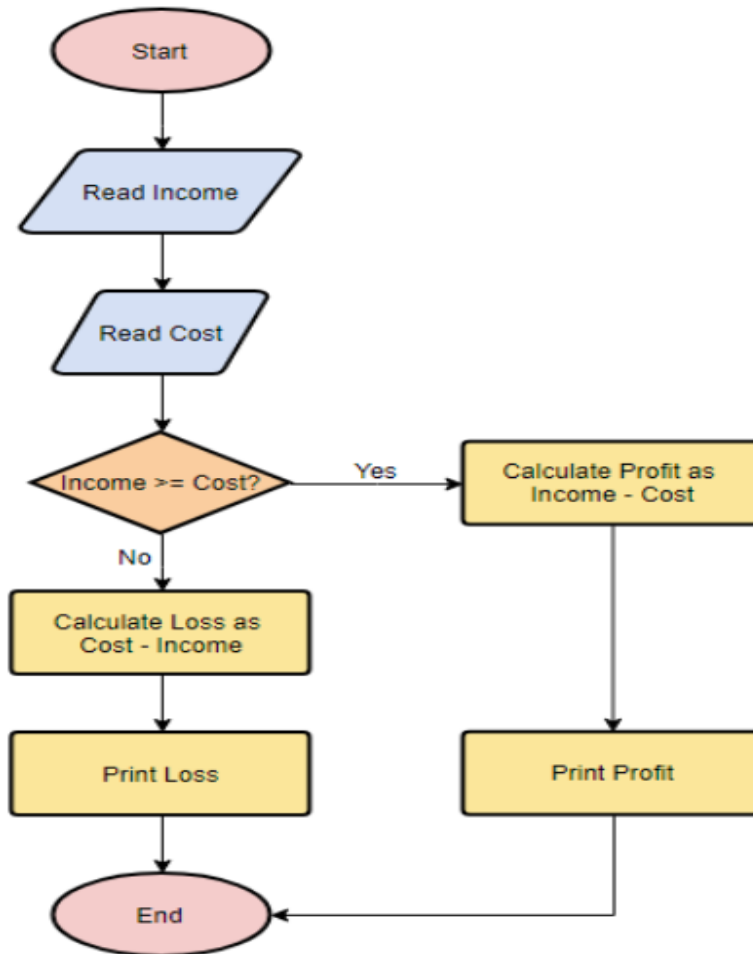
Find the sum of 529 and 256



# Flow-Chart Example

## Flowchart Example – Calculate Profit and Loss

Find the profit/loss when  
income = 1,000, cost = 800



# Analysis of Algorithms

---

1. Best Case
2. Average case
3. Worst Case



# Design of Algorithms

---

1. Divide-and-Conquer
2. Greedy
3. Dynamic Programming
4. Backtracking & Branch-and-Bound

