



CS 2022 : DATA STRUCTURES & ALGORITHMS

Lecture 1: Introduction

Malaka Walpola

OUTLINE

- ✿ Class Details
- ✿ Learning Outcomes
- ✿ Introduction
- ✿ Insertion Sort
- ✿ Bubble Sort



CLASS DETAILS

CS 2022 : DATA STRUCTURES & ALGORITHMS

- * Credits: 2.5 (GPA)
- * Pre-requisites : None
- * Course Objective:
 - * Provide an understanding of how to approach problem solving in computer science

CS 2022 : DATA STRUCTURES & ALGORITHMS

- ✿ Lecturers: Malaka Walpola
 - ✧ Contact information
 - ✧ Room: Inside the staff area of the CSE dept.
 - ✧ E-mail: malaka@cse.mrt.ac.lk
 - ✧ Phone: 0718661380

CS 2022 : DATA STRUCTURES & ALGORITHMS

* Hours/ Week:

- * Lectures – 2 hrs

 - * Thursday 9.15 - 11.15

- * Tutorial/Lab

 - * Monday 3.15 to 6.15

- * Reading the Book, Self Study & Homework: 5 hrs (will depend on individuals)

LEARNING OUTCOMES

- ✿ After completing this module, students should be able to
 - ✿ analyze the complexity of algorithms
 - ✿ implement and use common data structures
 - ✿ select appropriate data structures and algorithms for a given situation
 - ✿ apply basic algorithm design techniques for a given situation

COURSE OUTLINE

- * Complexity Analysis of Algorithms
- * Recursion
- * Searching
- * Sorting
- * Basic Algorithm Design Techniques
 - * Divide-and-conquer
 - * Greedy approach
 - * Dynamic programming

COURSE OUTLINE

- ✿ Basic Data Structures and Operations on Them
 - ✱ Arrays, Linked lists, Queues, Stacks
 - ✱ Sets
 - ✱ Trees
 - ✱ Hash tables
 - ✱ Graphs
- ✿ Introduction to NP-Completeness

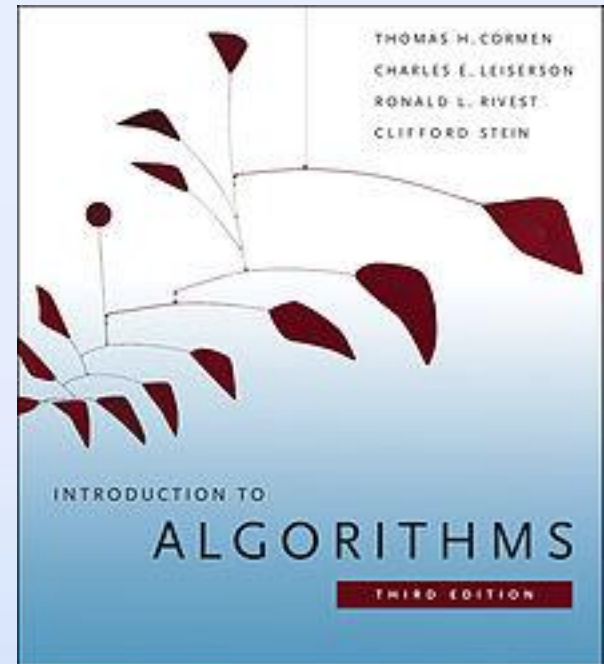
* Not in Order

METHOD OF ASSESSMENT

- ✿ Exam – 60% (35% to Pass)
 - ✿ 2 hour
 - ✿ Closed book
 - ✿ Answer all
 - ✿ Will have short answer/multiple choice questions
- ✿ Continuous Assessments – 40% (35% to Pass)
 - ✿ Labs – 60%
 - ✿ Mid semester exam - 40%

RECOMMENDED TEXT BOOKS

- ✿ Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, *Introduction to Algorithms*, 3rd Ed. Cambridge, MA, MIT Press, 2009.



CLASS RULES

- ✿ Be Respectful and Responsible
 - ✧ No talking among yourselves
 - ✧ No cell phones
- ✿ Be Organized and Follow Directions
- ✿ Be Prepared
 - ✧ If you miss a lecture, it is your responsibility to makeup
 - ✧ No makeup labs/tutorial slots

CLASS RULES

- * Take Home CA Must Be Your Own Work
 - * It will help you understand the material
 - * Cite all references
 - * We may be using plagiarism detection tools
- * Homework and In-class activities are Individual Work
 - * No peeking, copying, talking during In-class activities
 - * Don't let someone else copy from you as well
- * All course contents will be provided through the course Moodle page

EXPECTATIONS

- ✿ All the students are required to read the assigned sections of the book
 - ✿ Please keep up with the reading
 - ✿ In-class activities and homework will assume that you have done this.
- ✿ All the students are expected to actively participate in the in-class activities



QUESTIONS OR CONCERNS?



INTRODUCTION

LEARNING OUTCOMES

- ✧ After successfully studying contents covered in this lecture, students should be able to,
 - ✧ explain what an algorithm is and express an algorithm using pseudo code or flowcharts
 - ✧ explain the insertion sort, bubble sort & optimized versions of bubble sort

INTRODUCTION

- * What is an Algorithm?
 - * Well defined procedure
 - * Takes some inputs
 - * Produce some outputs
- * An algorithm is a ***step-by-step*** method of ***solving a computational task***
- * Why Do We Study Algorithms?
 - * To make understanding and solving problems simple
 - * To solve problems in the BEST way

SAMPLE PROBLEMS

- ✿ Searching the Web for the Stanford University Honor Code
- ✿ Calculating the Fourier Transform of a Signal
- ✿ Designing a PCB Layout for a Circuit
- ✿ Sorting a List of Names
- ✿ Calculating Best Path from Colombo to Anuradhapura

SAMPLE PROBLEMS

- ✿ Searching the Web for the *Stanford University Honor Code*
- ✿ Calculating Best Path from *Colombo* to *Anuradhapura*
- ✿ Calculating the Fourier Transform of a Signal
- ✿ Designing a PCB Layout for a Circuit
- ✿ Sorting a List of Names

COMPUTATIONAL TASK

- ✿ A computational task is not just a “single” task such as
 - ✿ “Is 3962431 prime ?”
 - ✿ “What is 37487×2371 ?”
- ✿ A computational task is a whole family of “similar” tasks with varying INPUT, such as
 - ✿ “Given a whole number A, is A prime?”
 - ✿ “Given 2 numbers x and y, what is x times y?”

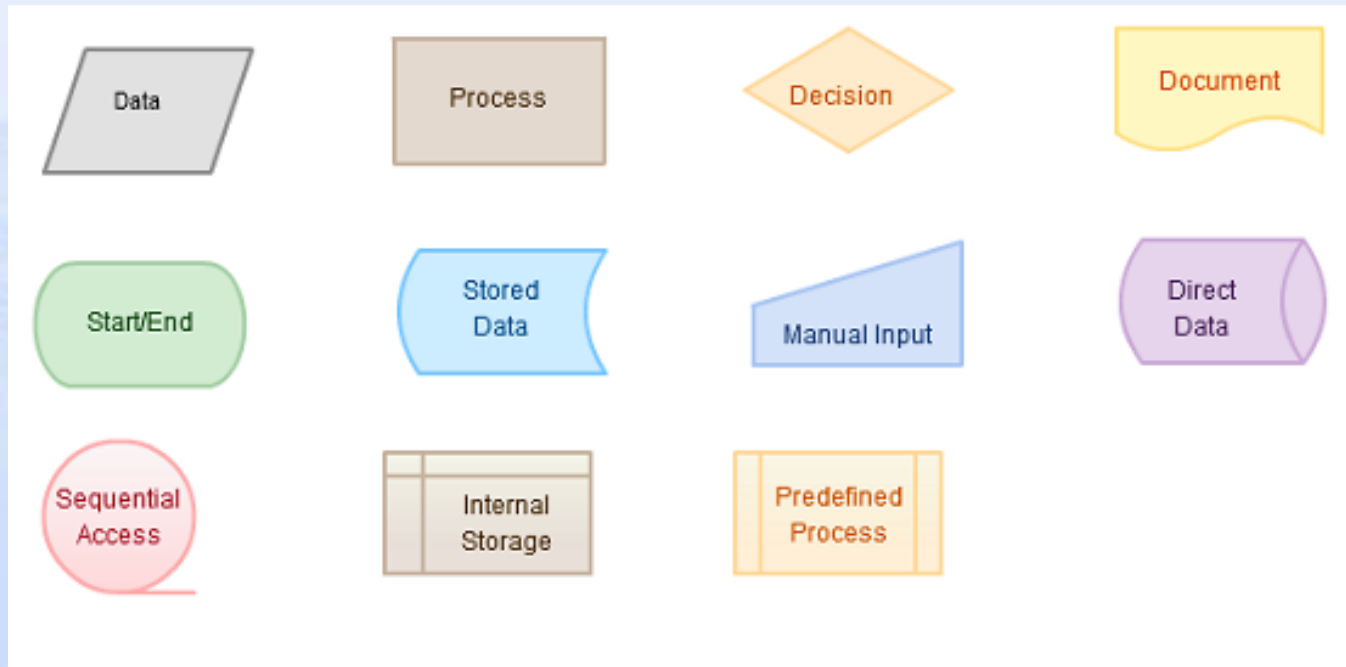
SPECIFYING ALGORITHMS

- ✿ Listing the Steps
- ✿ Flowcharts
- ✿ Pseudo Code
- ✿ Program Listing

- ✿ Class Activity
 - ✿ Specify an algorithm to add all the numbers in an array/list.

FLOWCHARTS

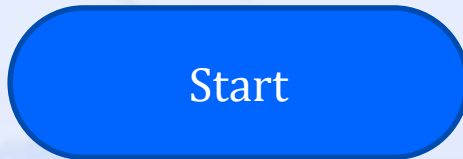
- ✿ A diagram that show the “flow of control” of an algorithm/a program
- ✿ Flowchart symbols



FLOWCHART SYMBOLS

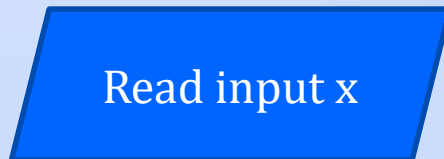
* Terminals

- * Represents the start or end of the process
- * Represented by rounded rectangles



* Input/output Data

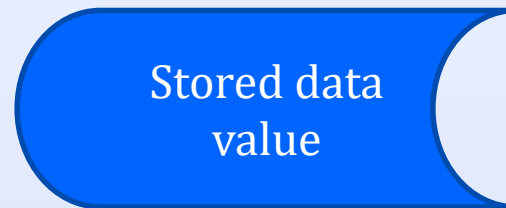
- * Represented by parallelograms
- * Indicate an input or output operation



FLOWCHART SYMBOLS

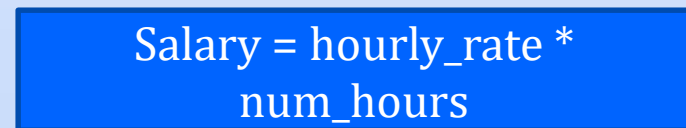
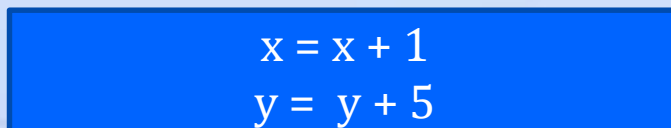
* Stored Data

- * Represented by rectangles
- * Indicates a process such as a mathematical computation or variable assignment



* Processing

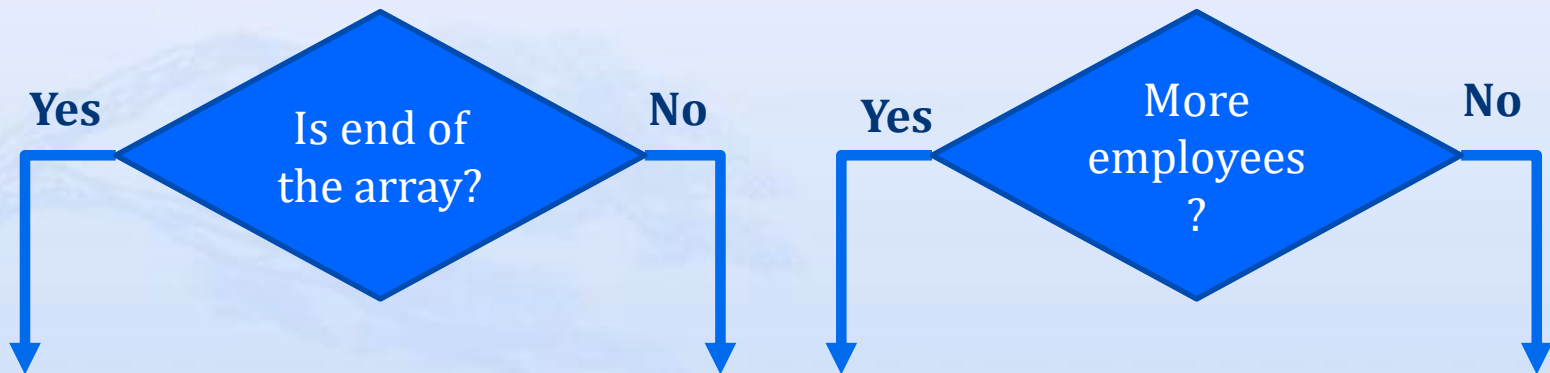
- * Represented by rectangles
- * Indicates a process such as a mathematical computation or variable assignment



FLOWCHART SYMBOLS

* Decision

- * Represented by diamond shape
- * Indicates different paths of execution/choices the program can take





INSERTION SORT

THE SORTING PROBLEM

- * Input:

- * A sequence of n numbers $\langle a_1, a_2, a_3, \dots, a_n \rangle$

- * Output:

- * A permutation (ordering) $\langle a'_1, a'_2, a'_3, \dots, a'_n \rangle$ of the input sequence such that $a'_1 \leq a'_2 \leq a'_3 \leq \dots \leq a'_n$

- * A More General Version

- * Input:

- * Output:

*Source [1]

INSERTION SORT

- ✿ Suitable for Sorting Small Number of Elements
- ✿ The Idea
 - ✿ Remove an element from input and insert it in proper location

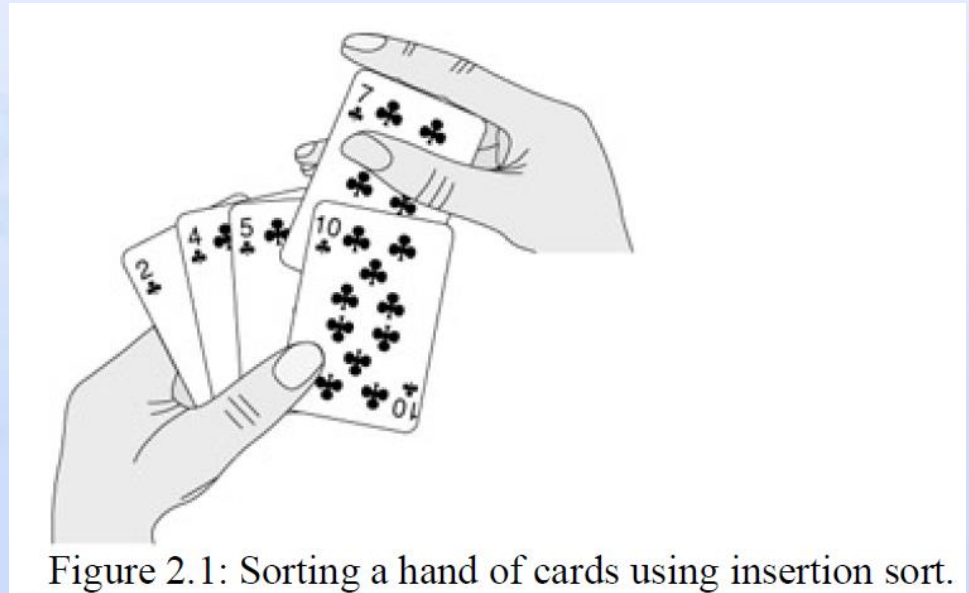


Image taken from [1]

Figure 2.1: Sorting a hand of cards using insertion sort.

INSERTION SORT

* The Idea

Sorted partial result		Unsorted data	
$\leq x$	$> x$	x	...

becomes

Sorted partial result		Unsorted data	
$\leq x$	x	$> x$...

Image taken from [3]

THE ALGORITHM

INSERTION-SORT (A)

```
1. for j = 2 to A.length
2.   key = A[j]
3.   //Insert A[j] into the sorted
   sequence A[1, ....., j-1].
4.   i = j-1
5.   while i > 0 and A[i] > key
6.     A[i+1] = A[i]
7.     i = i-1
8.   A[i+1] = key
```

BUBBLE SORT

* The Idea:

- * Bubble up the largest(smallest) element to the bottom(top) of the list in each iteration
- * Repeatedly step through the list
 - * Compare each pair of adjacent items
 - * Swap them if they are in the wrong order
 - * Repeated until no swaps are needed, which indicates that the list is sorted

THE ALGORITHM (HOMEWORK)

BUBBLE-SORT (A)

```
1.  do
2.      swapped = false
3.      for i = 2 to A.length
4.          if A[i-1] > A[i]
5.              temp = A[i]
6.              A[i] = A[i-1]
7.              A[i-1] = temp
8.              swapped = true
9.  while swapped
```

REFERENCES

- [1] T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein, *Introduction to Algorithms*, 3rd Ed. Cambridge, MA, MIT Press, 2009.
- [2] Wikipedia , “Insertion sort”, Oct. 09, 2012. [Online].
http://en.wikipedia.org/wiki/Insertion_sort. [Accessed: Oct. 22, 2012].
- [3] Wikipedia , “Bubble sort”, Oct. 17, 2012. [Online].
http://en.wikipedia.org/wiki/Bubble_sort. [Accessed: Oct. 22, 2012].

HOMEWORK

* Reading for Today's Class

- * Introduction: Chapter 1
- * Insertion Sort: Sections 2.1 & 2.2
- * Bubble Sort: [3]

* For Next Class

- * Asymptotic Notation: Section 3.1