Data Structures and Algorithms

Dr. Truong Dinh Huy

My Self

- Dr. Truong Dinh Huy
- Office: **Room 389**, Admin Building
- E-mail: huy.td@vgu.edu.vn
- Major: IoT, Deep Learning for IoT, Data Science
- Office hours: every Tuesday

Course Content

1. Asymptotic Analysis
2. Data Structures:
   - Linear data structures: List, Stacks, Queues
   - Non-linear data structures: Tree, Graph
3. Algorithms:
   - Basic: Sorting, Searching, Hashing
   - Advance: Graph Algorithms (Depth first Search, Breadth First Search, Shortest-Path algorithms,…)
4. Algorithm Design Techniques:
   - Greedy Algorithm
   - Divide and Conquer
   - Dynamic Programming
   - …

Course Workload

Student workload

Credits	5	ECTS
Contact hours	60	AHs
Assignments and independent learning	90	AHs
Total Working hours	150	AHs

Frequency The module is offered each academic year

Prerequisites
None

Assessment

1. 4 online Assignments: 40%
2. Final Exam (90 minutes written exam): 60%. Printed materials are allowed in exam room

Notes:
1. 4 Assignments will be done at lecture hall and online. You will use your laptop to do your assignments and submit them in our ILIAS website (elearning.vgu.edu.vn) in a limited time.
2. Prerequisites: Students are allowed to take part in the module examination if the student attended at least 75% of total number of lectures.
3. For students who do not follow this rule like 2020-students or 2019-students: To be allowed to take part in the final Exam, you must do all homework and submit them to specific folder named Submitted Homework.

Daily Schedule (expected)

Date	Location	Topic	Date	Location	Topic
05/03 1pm-4pm	Lecture Hall	Introduction	03/05 9am-12am	Lecture Hall	Graph
12/03 9am-12am	Lecture Hall	Analysis of Algorithm	10/05 9am-12am	Lecture Hall	Graph (2)
19/03 9am-12am	Lecture Hall	Sorting	17/05 9am-12am	Lecture Hall	Online Assignment 3 & Hashing
25/03 9am-12am	Lecture Hall	Searching	24/05 9am-12am	Lecture Hall	Greedy Algorithm
02/04 1pm-4pm	Lecture Hall	Online Assignment 1 & Linked List	31/05 9am-12am	Lecture Hall	Divide and Conquer
09/04 1pm-4pm	Lecture Hall	Stack, Queue	07/06 9am-12am	Lecture Hall	Online Assignment 4 &Dynamic Programming
16/04 1pm-4pm	Lecture Hall	Tree	14/06 9am-12am	Lecture Hall	Course Review
23/04 1pm-4pm	Lecture Hall	Online Assignment 2 &			

Materials

Recommended Textbook:

1. Data Structures and Algorithm Analysis in C, Mark Allen Weiss
2. Introduction to Algorithms, CLRS

E-learning platform: https://elearning.vgu.edu.vn/

1. Slides
2. HomeWork
3. Assignments

Course Motivation

Need to write computer programs efficiently!

Computer program:

- Accepts Input (Data)
- Performs a Sequence of action with the input
- Generates Output (Data)

How?

Efficient Management of Data

- Data Structures

Efficient Sequence of Actions

- Algorithms

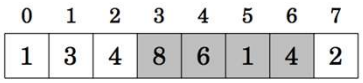
the efficiency of your algorithm:

- Run time
- Storage required

There is usually a trade-off between runtime and storage required

Example

- All these structures are there to efficiently store and process data
- Problem: Because we need the sum of a subarray of array **many times**, we will need to write a function running quickly:

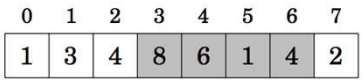


$sum_q(3, 6) = 19$

9

Example

- All these structures are there to efficiently store and process data
- Problem: find the sum of a subarray quickly:



$sum_q(3, 6) = 19$

Slower way ➔

```
int sum(int a, int b) {
    int s = 0;
    for (int i = a; i <= b; i++) {
        s += array[i];
    }
    return s;
}
```

10

Example

- All these structures are there to efficiently store and process data

- Problem: find the sum of a subarray of array A quickly:

0	1	2	3	4	5	6	7
1	3	4	8	6	1	4	2

$sum_q(3, 6) = 19$

Auxiliary Prefix Sum Array P:

0	1	2	3	4	5	6	7
1	4	8	16	22	23	27	29

Cooler/faster way ➔

P[i] = A[0] + A[1] +… + A[i]

11

Example

- All these structures are there to efficiently store and process data

- Problem: find the sum of a subarray quickly:

0	1	2	3	4	5	6	7
1	3	4	8	6	1	4	2

Cooler/faster way ➔

$sum_q(3, 6) = sum_q(0,6) - sum_q(0,2) = 27 – 8.$

Auxiliary Prefix Sum Array:

0	1	2	3	4	5	6	7
1	4	8	16	22	23	27	29

12

Example

- All these structures are there to efficiently store and process data

- Problem: find the sum of a subarray quickly.
- How to compute Prefix Sum Array P?

A:

0	1	2	3	4	5	6	7
1	3	4	8	6	1	4	2

P:

0	1	2	3	4	5	6	7
1	4	8	16	22	23	27	29

13

Example

- All these structures are there to efficiently store and process data
- Problem: find the sum of a subarray quickly.
- How to compute Prefix Sum Array P?
  - Dead simple application of dynamic programming:
    - `P[0]=A[0]; for(i=1 to n-1) P[i]=P[i-1]+A[i];`

0	1	2	3	4	5	6	7
1	3	4	8	6	1	4	2

0	1	2	3	4	5	6	7
1	4	8	16	22	23	27	29

Disadvantage: We need some memories to store array P

14