Data Structures

Notes from
Dr. Charlie Obimbo and Dr. Andrew Hamilton-Wright

Revised by Yan Yan.

Contents

1. Data Structures
2. Some Data Structures & their Applications

Data Structures

- Data Structures - specialized means of organizing and storing data in computers in such a way that we can use and perform operations on the stored data efficiently.
  - Data = = Information?
  - Let’s see an example
  - Data: nrael atad erutcurts
  - Information: learn data structure

Data Structures

- Data Structures have a wide and diverse scope of usage across the fields of Computer Science and Software Engineering.

- No data structure is generic or can be suitable for all purposes, so it is important to know their advantages and limitations.

CIS*2520 Learning Outcomes

- **Describe** and **implement** common data structures for solving complex programming problems including algorithms for the creation, insertion, deletion, searching, and sorting of each data structure.

- **Analyse** the space and time efficiency of algorithms including algorithms for the creation, insertion, deletion, searching, and sorting of data structures discussed.

- **Select** and correctly **use** the appropriate abstract data type for programming problems.

Space and Time Efficiency

- What does it mean to “be **efficient**”?

- Efficiency is about the use of *resources*
  - we can measure it: fewer resources means higher efficiency (sometimes described as “being more efficient”)

- As a running program, we effectively have only two resources: time and space

Space and Time Efficiency

- Time : the time taken up before a task completes

- Space : the amount of storage required for the bytes of data used by the running program — either
  - memory(RAM) – the bytes immediately available to be used in operations on the CPU
  - disk/device storage – the bytes stored persistently on external slower devices

Space or Time

- Space/time trade off
  - In some cases, people “buy time” by “spending space”

- We may be able consider other strategies that are better in space **and** time.
  - cleverly designed *algorithms* that need fewer operations
  - cleverly designed *storage strategies* that get more information into each byte

What is the Best?

- Why not just use the cleverest strategy all the time?

- The context that makes one strategy efficient may make another one very wasteful

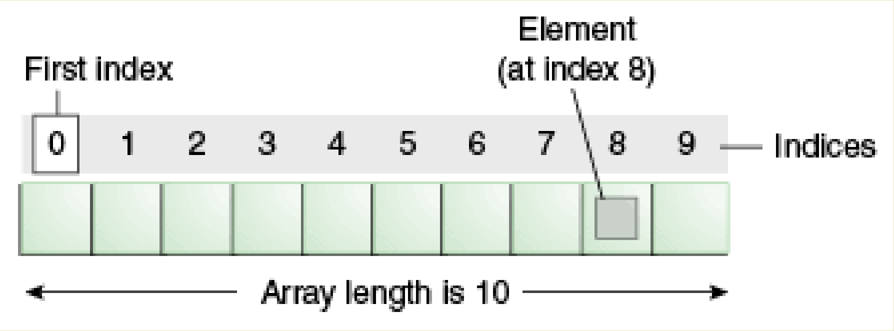
- In this course, we introduce you to these strategies and you make **good choices** because you can recognize
  - which one will help you in a given context
  - why the best choice in a given context is the best

Contents

1. Data Structures
2. **Some Data Structures & their Applications**

Some Data Structures & Applications

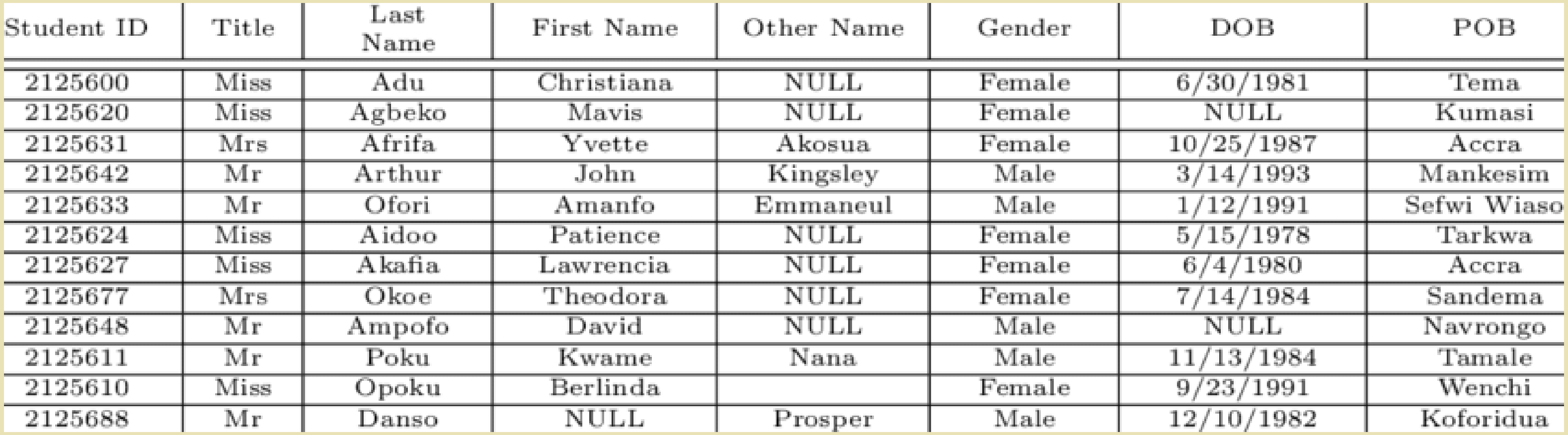
- Array:
  - a container object that holds a fixed number of values of a single type. The length of an array is established when the array is created.



  - Why use array?
    - Create one single structure to store multiple objects of the same type (for operations)

Some Data Structures & Applications

- Applications of Array:
  - – To store multiple elements of the same data type or data structures, e.g. students records:



Student ID	Title	Last Name	First Name	Other Name	Gender	DOB	POB
2125600	Miss	Adu	Christiana	NULL	Female	6/30/1981	Tema
2125620	Miss	Agbeko	Mavis	NULL	Female	NULL	Kumasi
2125631	Mrs	Afrifa	Yvette	Akosua	Female	10/25/1987	Accra
2125642	Mr	Arthur	John	Kingsley	Male	3/14/1993	Mankesim
2125633	Mr	Ofori	Amanfo	Emmaneul	Male	1/12/1991	Sefwi Wiaso
2125624	Miss	Aidoo	Patience	NULL	Female	5/15/1978	Tarkwa
2125627	Miss	Akafia	Lawrencia	NULL	Female	6/4/1980	Accra
2125677	Mrs	Okoe	Theodora	NULL	Female	7/14/1984	Sandema
2125648	Mr	Ampofo	David	NULL	Male	NULL	Navrongo
2125611	Mr	Poku	Kwame	Nana	Male	11/13/1984	Tamale
2125610	Miss	Opoku	Berlinda		Female	9/23/1991	Wenchi
2125688	Mr	Danso	NULL	Prosper	Male	12/10/1982	Koforidua

Some Data Structures & Applications

- Applications of Array:
  - Sorting Elements or records;
    - Using Quick sort
    - Heap sort, etc.
  - CPU Scheduling (Queues)
  - Can be used in Recursive functions

Some Data Structures & Applications

- **Stacks & Queues**
  - Stacks: Last in First Out

  - Queues: First in First Out

- Stack - a list in which all insertions and deletions are made at one end, called the TOP. The last element to be inserted into the stack will be the first to be removed.
  - Applications
    - Page-visited history in a Web browser
    - Undo sequence in a text editor
    - Saving local variables when one function calls another, and this one calls another, and so on.
  - Indirect Applications
    - Auxiliary data structure for algorithms
    - Component of other data structures

- Queue – deletion is done at one end (called the front of the queue) and into which items may be inserted at the other end (the rear of the queue)..
  - Applications
    - Waiting lines
    - Access to shared resources (e.g., printer)
    - Multiprogramming
  - Indirect Applications
    - Auxiliary data structure for algorithms
    - Component of other data structures.

Some Data Structures & Applications

- **Graphs**
  - Collections of vertices (notes) connected by edged (line segments)
  - Show relationships

between data

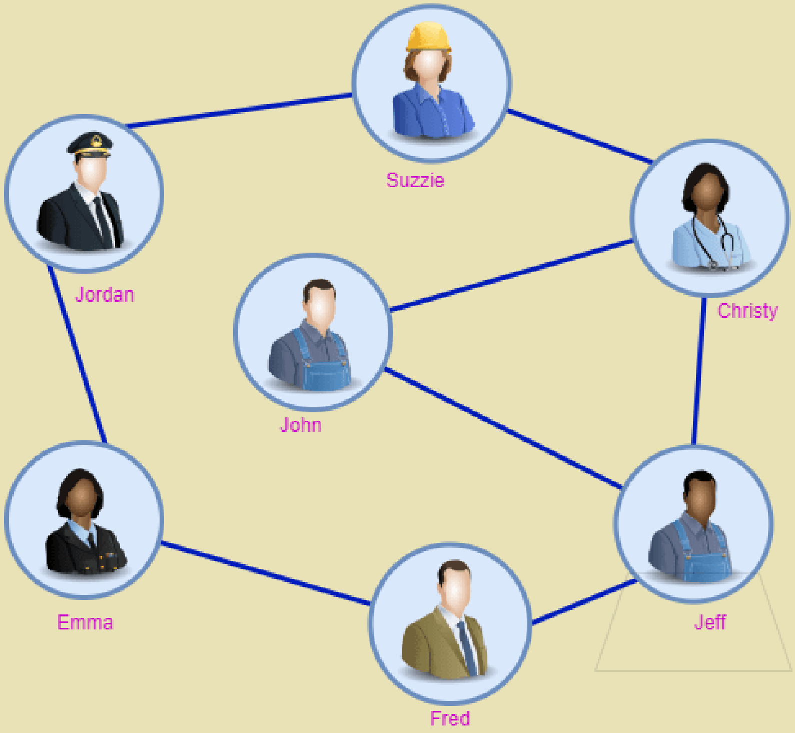


Image from https://dev.to/m13ha/data-structures-graphs-i-3mdg

Social Media Graph