Data Structures

session

Tony Stone

Contents

1 Preface

1.1 Mission Statement

With this work, I set out to develop a comprehensive set of lecture notes, on the topic of Data Structures. While developing these notes, I am keeping in mind both the lecturer, and the learner. My goal is that this will amount to an educational resource that acts as a guide throughout fundamental topics, in a manner which is accessible and painless for all parties. This work should provide confidence and direction to its reader, whether they are at the front of a classroom, or at their dining-room table.

1.2 Acknowledgments

I would like to thank the Center for Academic Success and Accessibilty Services at Southern Connecticut State University, for the employment and opportunity to develop these materials. I would like to extend this thank you for all of the administrative support I have recieved to my boss Kathleen De Oliveira, and coordinaters Tanisha Guadalupe De Jésus, Calleigh Cotter, and Mia Grella. Of course, I would like to extend this thank you to Professor Carl Haberfeld, for allowing me to attend his CSC-212 lectures.

2 Introduction's Introduction

I am very excited to dive into the wonderful world of **data structures**. Before we get started, there is just a bit of housekeeping to do. These matters are more relevant to the themes of this work, and the text as it appears on paper. I figure this is not really an introduction to data structures, its more of a introduction to data structures's introduction!

2.1 Python Examples

Topics and key ideas will usually correspond to some kind of Python example. An assumption is being made that the reader is to some extent *aquainted* with Python. There is absolutely no need to be a wizard of Pythonism, so there are notes clarifying syntax where necessary.

2.1.1 I Hate Snake Case

Per the title of this soap-box, I am not a fan of 'snake_case' as a naming convention. For the Python examples I will be writing in 'camelCase.' Is it my early Java Indoctrination peaking through? Maybe—but a set naming convention is almost entirely superfluous, and it looks nicer to my eyes. camelCase it is.

Let's get accustomed to it, here's an Example!

(`Example01.py`)

```
1 def sayHi():
2     print("Hello world!")
3
4 sayHi()
```

The reference to the actual script 'Example01.py' is included for your convenience, it can be found in the corresponding GitHub Repository!

2.1.2 Further Reading

Many of the terms and concepts featured in here will be based on the textbook *Problem Solving with Algorithms and Data Structures using Python* by Brad Miller and David Ranum. They have a fantastic online resource, so please check it out!

- https://runestone.academy/ns/books/published/pythonds/index.html

3 Introduction to Data Structures

3.1 Defining Data

Lets begin with getting the terms out of the way, and associating them with a defintion.

- **Primative data type**: A definition of data, with regards to allowable values, and operations.
- **Abstract data type** (ADT): A logical, abstract definition of how data is to be viewed.
- **Data structure**: The actual implementation of an ADT in a given programming language.

These terms at first glance seem like sticking 'data' next to a bunch of other computer-y words. Some of their coonnections and relations can be muddy. So let's take a moment to try to understand them through some exxplanation, and examples.

3.1.1 Primative Data Types

The **primative data types** include data such as *integers*, *floating point numbers*, and *booleans*. These are primative (**sometimes called atomic**) because they are the small divisions of data, that are *used* to make up data structures. As given in the above definition, they are defined by the possible values and the possible operations they can participate in.

3.1.2 Abstract Data Types

Abstract data types are instead defined by *behavior*, what it does. We will look into a data structure called a **Stack**. For now, know that this is a collection, and we define it as a collection which releases objects in reverse-order of their arrival. Note: I did not say this was a collections of any kind of thing, or had any specific operations. For ADTs, we're concerned with how they act!

3.1.3 Data Structures

Data structures are the real deal. A data structure is the actual implementation of an ADT which we will interface with. Back to our stack, we might use a standard python list, and only append and pop items. This is a very basic example, with some data-structures later, we will be giving it more effort than this!

But with those definitions, lits now take a look at a python script, and identify what-is-what:

(Example02.py)

```python
# Primative Data Type
print("Im an integer, called ", type(6))

# Abstract Data Type
print("I want to release data in reverse order of it's
    arrival")

# Data Structure
stack = [1, 2, 3]
stack.append(4)
print(f"4 comes and goes: {stack.pop()}, leaving {stack}")
```

References

[1] Bradley N. Miller and David L. Ranum. *Problem Solving with Algorithms and Data Structures Using Python.* Franklin, Beedle & Associates, Wilsonville, OR, 2 edition, 2011.