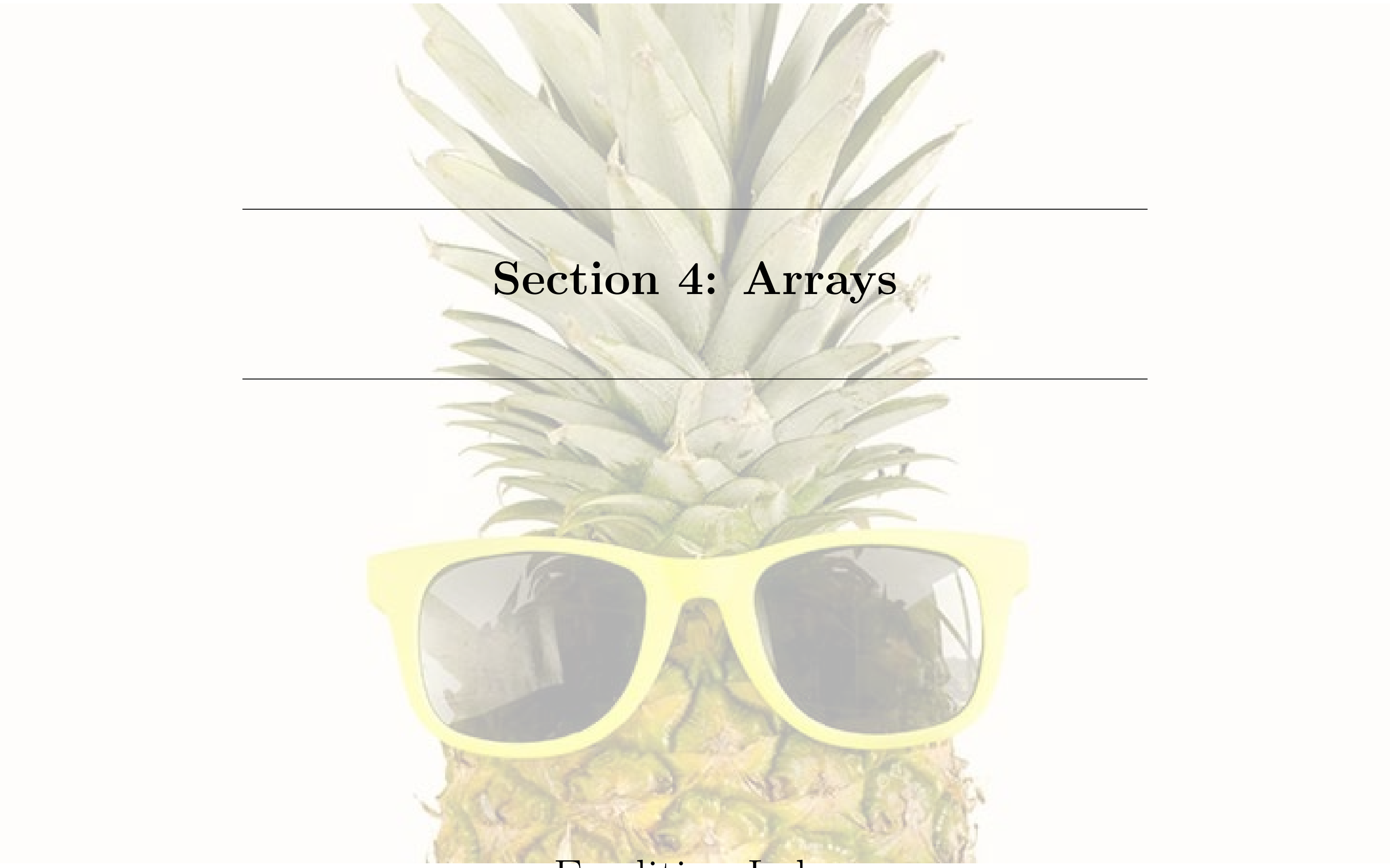

Section 4: Arrays

Erudition Labs

Computer Science 101: Introduction to Java and Algorithms

May 8, 2019

Contents

1 Terminology

- ***Memory*** – memory is
- ***Data Structure*** –
- ***Collections*** –
- ***Array*** –
- ***Elements of an Array*** –

2 Pre-Chapter

2.1 Heap and Stack Memory(Over Simplified)

2.2 The “new“ keyword

2.3 Declaration vs. Initialization

3 Arrays (Video Series Lecture 20 and 21)

This section introduces Arrays, which is a data structure that stores collections of elements. What is a data structure? Well a data structure is just a way organizing data by following certain rules when we store and retrieve it. By doing this, we can take advantage of the way the data is organized to make efficient algorithms. Some data structures are better at some things than others. It all depends on what you need to do. So when you take a data structures class (or look into it yourself) you will learn about different data structures (aka different ways of organizing data), their rules, their advantages, disadvantages and their best use cases.

For example, if your program is going to be doing a lot of retrieving, there are data structures that are great for that. If you need to do a lot of saving or insertions, we have great data structures for that too, but those might also not be the best for retrieving. There are often trade-offs when using them and it is up to the programmer to decide which ones are best for your task. For now, you don‘t really need to know what a data structure is other than it‘s a way if organizing data. If you are taking computer science classes, then there are entire classes dedicated to that topic. All you need to know right now is that an array is used to store a collection of things of the same type and are extremely good for random access (meaning I can get any arbitrary piece of data at any time efficiently as long as I know where it is at in the array).

3.1 What does an Array Look Like?

Arrays are contiguous chunks of memory. And they MUST be contiguous due to how we access the data. I will talk about that more later. Usually when we try to visualize and conceptualize what an array is, we draw it like this:

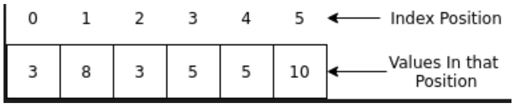

Figure 1: Image from Lecture

Imagine, if you will, having a bunch of variables of the same type under one name. This is essentially what an array is. So if you need to store a collection of integers, you can declare one array to store them instead of declaring a bunch of different variables. If you look at the table, the top row corresponds to the index position of the array. The index is simply where that particular piece of data is stored at in the array. Also note that we refer to these “pieces of data“ as elements. For example, if we look at the table above, we can see that 8 is an element of that array at index position 1.

Notice that we start counting from 0. The 0th element is actually the first element in the array. Next, you can see all the boxes (aka the elements). Think of each box as a variable that you don‘t have to name that stores a value. So in the image, we have an array of size 6 that stores 6 integers.

3.2 Declaring/Initializing an Array

We can declare an array just like we declare any other variable, in the form of

```
int myArray[];
```

4 Looping Over Arrays (Video Series Lecture 22 and 23)

5 2D Arrays (Video Series Lecture 24 and 25)

6 Sorting Arrays: Insertion Sort Algorithm (Video Series Lecture 26 and 27)

7 Gotchas with Arrays