Arrays and ArrayLists

Term 2 - Week 2 - Academic Year 2022-23

Arrays

- Recap
  - Array Data Structure
  - Linked List Data Structure
- Arrays
- ArrayLists
- The List Interface
- Discussion Point

Recap: Array Data Structure

- Immutable Data Structure
- Indexing has a time complexity of O(1)
- The default "list" in C! (disregarding malloc())

Recap: Linked List Data Structure

- Mutable Data Structure
- Indexing has a time complexity of O(n) with n the number of elements in the list.
- The default "list" in Python!

Arrays

- Immutable Data Structure. The size cannot be changed, but values can.
- Declaration and initialization can be done together or separately.

```
// Declaration
int array[];
int[] array2;
String[] song;

// Initialization
array = new int[]{1, 2, 3}; // `new datatype[]` is compulsory
song = new String[]{"Somebody's", "watching", "me"};
```

Arrays

```
// Declaration + Initialisation.
int[] array3 = {1, 2, 3}; // `new datatype[]` is optional
String[] song2 = {"Smells", "like", "teen", "spirit"};
String [] threeWords = new String[3]; // Preallocate size.
```

Arrays - Common Operations

- Declaration + Initialization

```
song = new String[]{"Somebody's", "watching", "me"};
```

- Indexing & Replacing

```
song[2] // Indexing
song[2] = "you" // Indexing + variable change.
```

- Getting the array's length

```
song.length // Get length of array
```

ArrayLists

- Java's mutable array!
  - Allows more functionality on top of existing array functions.
- Does not support **primitive types**.

```
// Separate declaration & initialisation
ArrayList<Integer> intList;
intList = new ArrayList<>();

// Combined declaration & initialisation
ArrayList<String> arrayList = new ArrayList<>();
List<String> list = new ArrayList<>();
```

ArrayLists - Common Operations

- Addition

```
ArrayList<String> confessions = new ArrayList<>();
confessions.add("Hung Up"); // Addition
confessions.add("Get Together");
confessions.add("Sorry");

ArrayList<String> celebration = new ArrayList<>();
celebration.add("Material Girl");
celebration.add("4 Minutes");
confessions.addAll(celebration); // Add multiple elements
```

ArrayLists - Common Operations

- Retrieval

```
confessions.get(0); // Element retrieval
```

- Deletion

```
confessions.remove(3); // Deletion via index
confessions.remove("Like a Prayer"); // Deletion via search
```

- Contains

```
confessions.contains("Sorry"); // Check if list contains element.
```

- Search

```
confessions.indexOf("Sorry"); // Search index of element.
```

- Length

```
confessions.size(); // Length of array.
```

The List Interface

- ArrayList is an implementation of the List interface
  - We will see interfaces later.
  - Essentially an implementation of a template.

- One can choose to store an ArrayList in a List variable.
  - Allows for more List implementations to be passed as arguments/returned despite Java's type checking.

Discussion Point

- ArrayLists have both the functionality of an array and a list. How do you believe this was achieved?
- This is a short session – but it's very important that the basics of Collections (a later class), ArrayLists, are understood!

Workshop

With only immutable Java arrays, create a mutable array. Do not use the ArrayList class.

If you haven't done so already, fork this repository to add your solution:

https://github.com/Catcatcher33/programming-tutor-22-23