

CSCI 132:

Basic Data Structures and Algorithms

ArrayLists

Reese Pearsall
Spring 2024

Announcements

- Program 1 due on Friday @ 11:59 PM

No office hours on Friday (1:30 – 2:30)

Why did the programmer quit his job?

He didn't get arrays.

Array Limitations

Cons

- **Can't change the length**
- Can only store one data type

What can we do about this?

```
int[] myArray = {1, 2, 3};  
System.out.println(Arrays.toString(myArray));
```

```
int[] newArray = new int[myArray.length + 1];  
for(int i = 0; i < myArray.length; i++) {  
    newArray[i] = myArray[i];  
}
```

// Create a new array that is one spot bigger

// Fill new array with contents of old array

```
int new_value = 4;  
newArray[myArray.length] = new_value;  
myArray = newArray;
```

// add new value to array

// Update reference variable

Array Limitations

Cons

- **Can't change the length**

Solution

Create new array, copy everything over
(this can be expensive ☹)

```
int[] myArray = {1, 2, 3};  
System.out.println(Arrays.toString(myArray));
```

```
int[] newArray = new int[myArray.length + 1];  
for(int i = 0; i < myArray.length; i++) {  
    newArray[i] = myArray[i];  
}
```

```
int new_value = 4;  
newArray[myArray.length] = new_value;  
myArray = newArray;
```

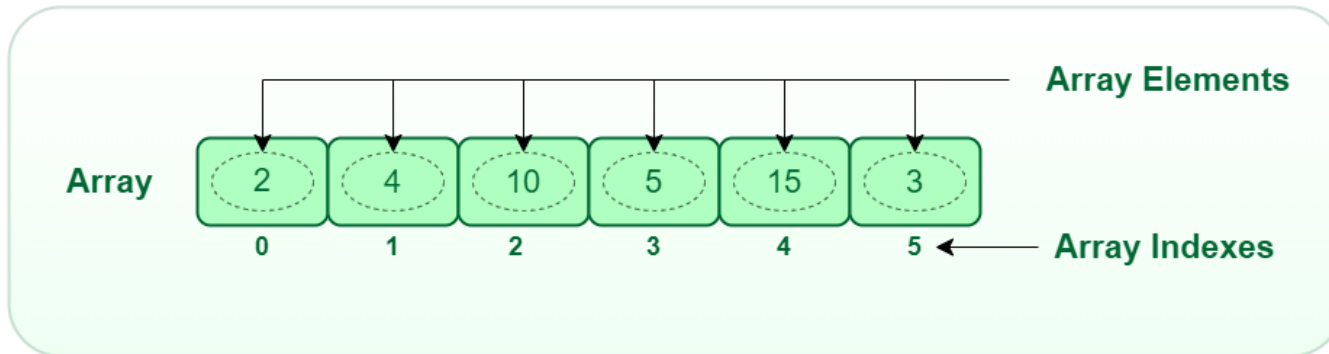
- **Can only store one data type**

Solution

Store an object, use two separate arrays, use a different data structure

An **ArrayList** is a data structure that can hold multiple, similar values (just like an array), **BUT**

- Dynamic, can easily resize
- Can easily add new elements and remove elements
- Like a Python list 😊



Somebody took `arrays`, and made them better

- Still have indices
- Still can only store one data type

Java ArrayLists

We first need to remember to import it ☺

```
import java.util.ArrayList;
```

Creating a new ArrayList

```
ArrayList<String> mylist = new ArrayList<String>();
```

We must specify
what datatype the
ArrayList will hold

Reference Variable

We never have to specify the
size, because we can now easily
grow/shrink our ArrayList!

Java ArrayLists

We first need to remember to import it ☺

```
import java.util.ArrayList;
```

Creating a new ArrayList

```
ArrayList<String> mylist = new ArrayList<String>();
```

We can add stuff to the ArrayList using the `.add()` method (built in method!)

```
mylist.add("Jack");
```

Java ArrayLists

We first need to remember to import it ☺

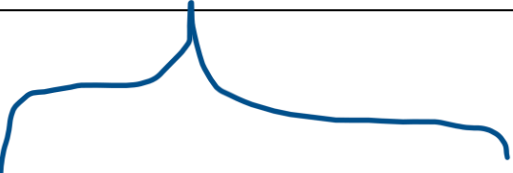
```
import java.util.ArrayList;
```

Creating a new ArrayList

```
ArrayList<String> mylist = new ArrayList<String>();
```

We can add stuff to the ArrayList using the `.add()` method (built in method!)

```
mylist.add("Jack");
```



```
int[] myArray = {1, 2, 3};
System.out.println(Arrays.toString(myArray));

int[] newArray = new int[myArray.length + 1];
for(int i = 0; i < myArray.length; i++) {
    newArray[i] = myArray[i];
}

int new_value = 4;
newArray[myArray.length] = new_value;
myArray = newArray;
```

Under the hood, it is

1. Creating a new array
2. Copy old contents
3. Add new element at the end
4. Updating reference variable

Java ArrayLists

We first need to remember to import it ☺

```
import java.util.ArrayList;
```

Creating a new ArrayList

```
ArrayList<String> mylist = new ArrayList<String>();
```

We can add stuff to the ArrayList using the `.add()` method (built in method!)

```
mylist.add("Jack");
```

To access elements in the array, we use the `.get()` method (we cannot use the square bracket index `[]`)

```
System.out.println(mylist.get(2)); // this will print the String at index 2
```

Java ArrayLists

We first need to remember to import it ☺

```
import java.util.ArrayList;
```

Creating a new ArrayList

```
ArrayList<String> mylist = new ArrayList<String>();
```

We can add stuff to the ArrayList using the `.add()` method (built in method!)

```
mylist.add("Jack");
```

To access elements in the array, we use the `.get()` method (we cannot use the square bracket index `[]`)

```
System.out.println(mylist.get(2)); // this will print the String at index 2
```

We can remove stuff by index, or by searching for a specific element

```
mylist.remove("Eli");  
mylist.remove(0);
```

Java ArrayLists

```
import java.util.ArrayList;
public class ArrayListDemo {

    public static void main(String[] args) {
        ArrayList<String> mylist = new ArrayList<String>();

        mylist.add("Jack");
        mylist.add("Tory");
        mylist.add("Sam");
        mylist.add("Eli");

        System.out.println(mylist);
        System.out.println(mylist.get(2));

        mylist.remove("Eli");
        mylist.remove(0);

        System.out.println(mylist);
        System.out.println(mylist.isEmpty());
    }
}
```

Java ArrayLists Example

Let's write a program that will keep track of high scores on an arcade machine

www.gamesdatabase.org

BEST PLAYERS

NO.	SCORE	NAME
1	36500	---BMB
2	34900	---RAD
3	33100	---P.P
4	31700	---FAL
5	29900	---ZIB
6	28300	---TOZ
7	27100	---LWP
8	25700	---JUF
9	24500	---TCZ
10	22900	---TCR

Each entry will have the player name (String), and their score (Int)

The program should allow for

- Adding a new high score
- Removing a score
- Print out scoreboard
- Print out top N scores
- Search for score by name

And we must use an **ArrayList** to hold all this information!