CSCI 132:
Basic Data Structures and Algorithms

Arrays

Reese Pearsall & Iliana Castillon

Fall 2024

Announcements

Program 1 due **Sunday** at 11:59 PM

Lab 4 due **Thursday** at 11:59 PM

```
Roses are Red,
Violets are Blue.

Unexpected ‘{’ on line 32.
```

What do you need to dig a hole?

What do you need to dig a hole?

What do you need to dig a hole?

	Pros	Cons

What do you need to dig a hole?

	Pros	Cons
	• Cheap • Precise • No Training • Availability	• Slow • Labor

Thank you to Sean Yaw (the goat) for the analogy

What do you need to dig a hole?

	Pros	Cons
	• Cheap • Precise • No Training • Availability	• Slow • Labor
	• Fast • Labor	• Expensive • Training

What do you need to dig a hole?

	Pros	Cons
	• Cheap • Precise • No Training • Availability	• Slow • Labor
	• Fast • Labor	• Expensive • Training
	• Really good at digging	• Takes up a lot of garage space

What do you need to dig a hole?

	Pros	Cons
	• Cheap • Precise • No Training • Availability	• Slow • Labor
	• Fast • Labor	• Expensive • Training
	• Really good at digging	• Takes up a lot of garage space

Best tool for the job?

Burying your pet goldfish

What do you need to dig a hole?

	Pros	Cons
	• Cheap • Precise • No Training • Availability	• Slow • Labor
	• Fast • Labor	• Expensive • Training
	• Really good at digging	• Takes up a lot of garage space

Best tool for the job?

Building Express tunnel to Bridger Bowl

What do you need to dig a hole?

	Pros	Cons
	• Cheap • Precise • No Training • Availability	• Slow • Labor
	• Fast • Labor	• Expensive • Training
	• Really good at digging	• Takes up a lot of garage space

Best tool for the job?

Creating the foundation for a house

What do you need to dig a hole?

	Pros	Cons
	• Cheap • Precise • No Training • Availability	• Slow • Labor
	• Fast • Labor	• Expensive • Training
	• Really good at digging	• Takes up a lot of garage space

Best tool for the job?

Digging a Well for water

What do you need to dig a hole?

	Pros	Cons
		w or
		ensive ning
		s up a lot of ge space

Best tool for the job?

Digging a Well for water

What do you need to dig a hole?

	Pros	Cons
		w or
		ensive ning
		s up a lot of ge space

We can't use the best tool for the job unless we know that tool exists!

Best tool for the job?

Digging a Well for water

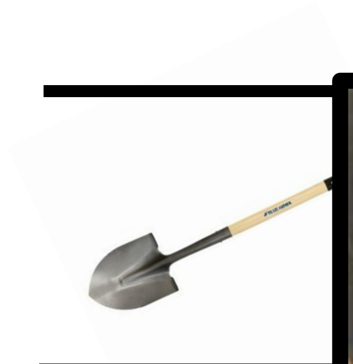
What do you need to dig a hole?

	Pros	Cons
	• Cheap • Precise • No Training • Availability	• Slow • Labor
	• Fast • Labor	• Expensive • Training
	• Really good at digging	• Takes up a lot of garage space

Best tool for the job?

Creating the foundation for a house

What do you need to dig a hole?

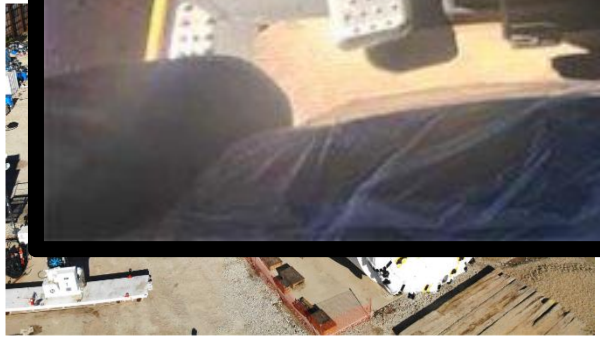
	Pros	Cons
		
		ve
		a lot of ace

Best tool for the job?

Creating the foundation for a house

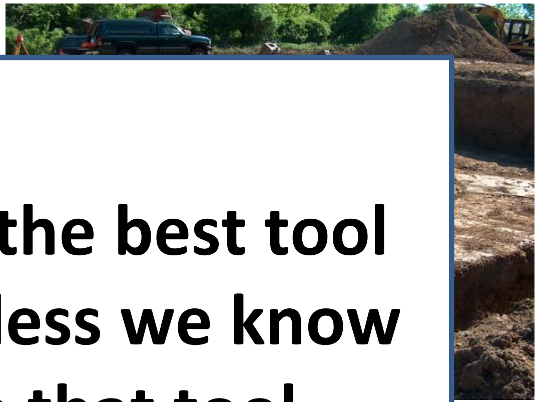
What do you need to dig a hole?

	Pros	Cons
		• Slow • Labor
		garage space



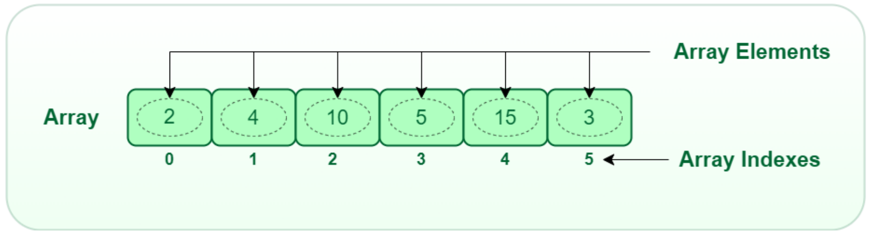
Best tool for the job?

Creating the foundation for a house

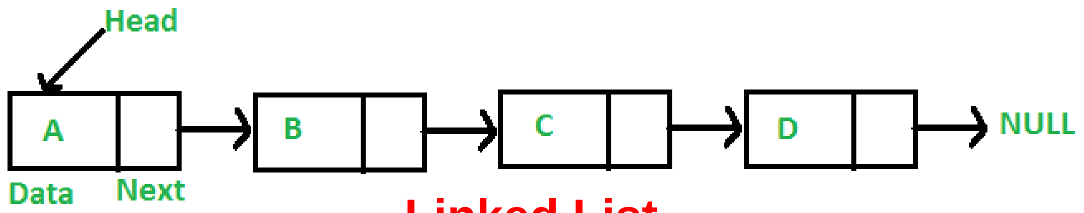


We can't use the best tool for the job unless we know how to use that tool

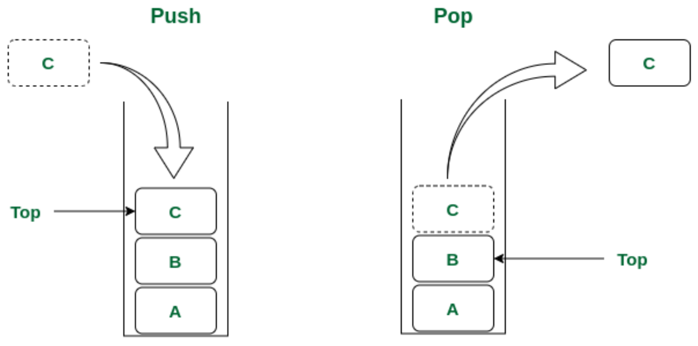
A **data structure** is a mechanism for storing and organizing data



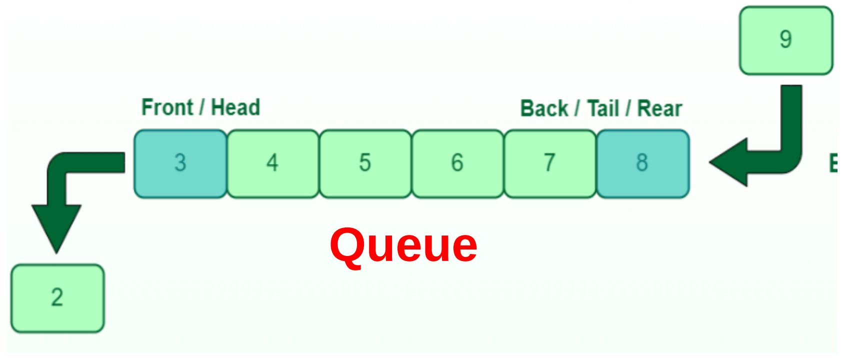
Arrays



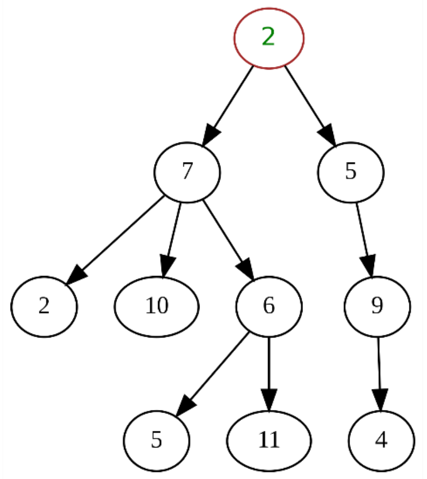
Linked List



Stack



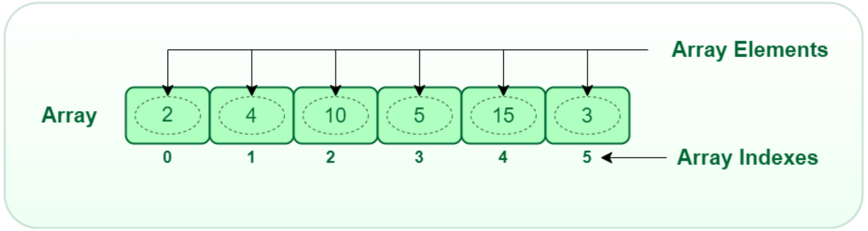
Queue



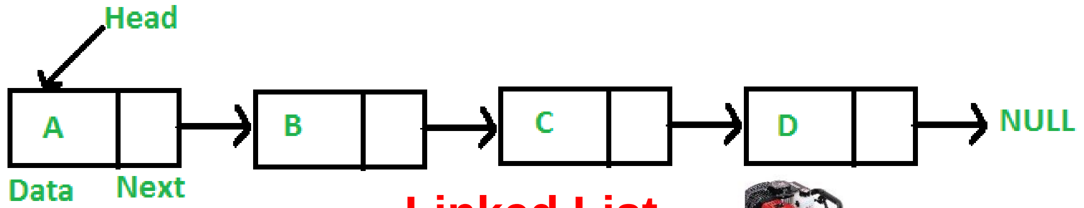
Trees

A **data structure** is a mechanism for storing and organizing data

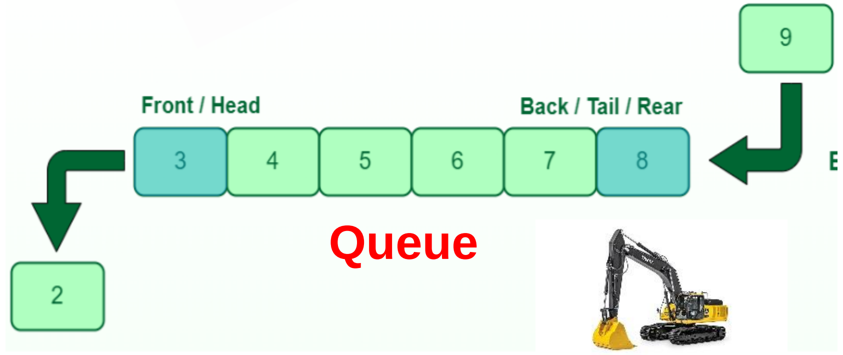
- We have structured ways of *accessing* and *managing* data



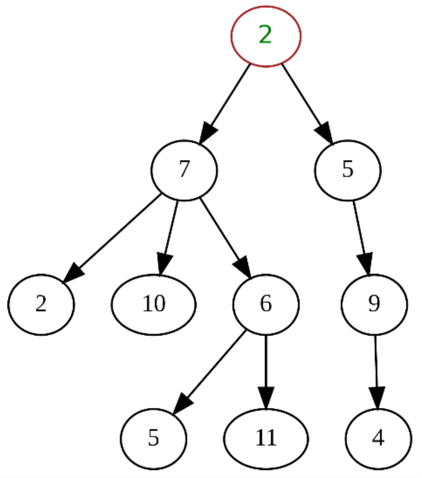
Arrays



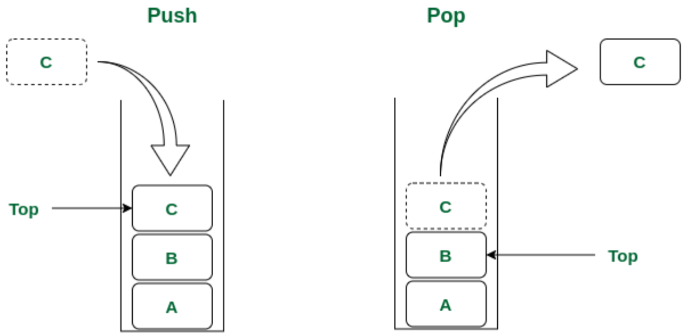
Linked List



Queue



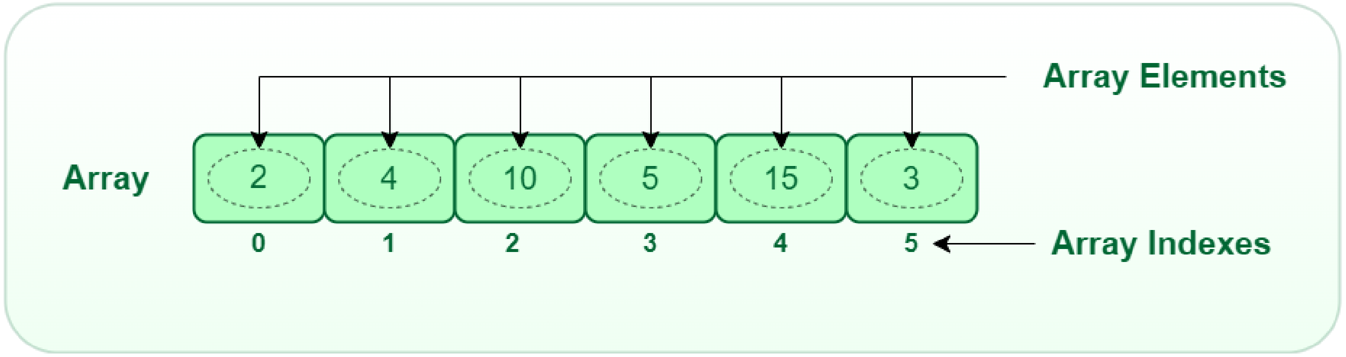
Trees



Stack

There are many types of data structure, and each data structure has its pros and cons

An **array** is a data structure that can hold multiple, similar values

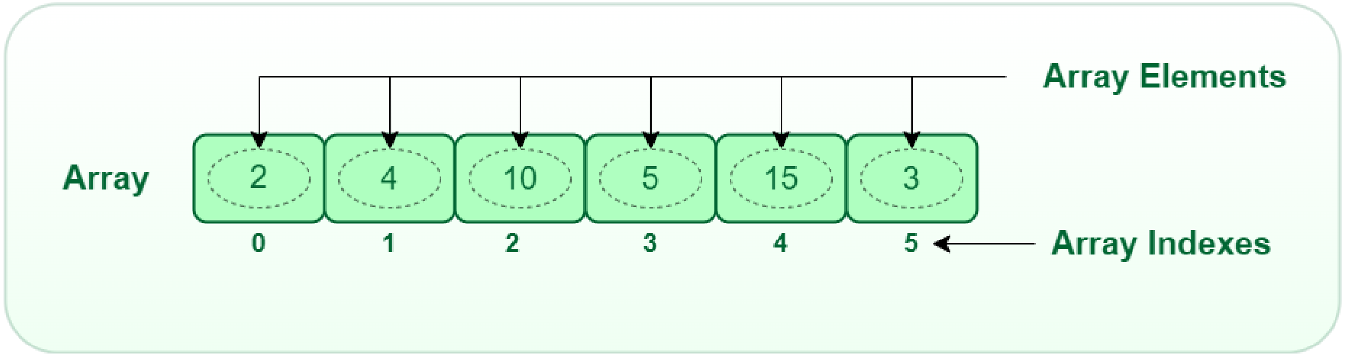


```
String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};

int[] myNum = {10, 20, 30, 40};
```

<u>Pros</u>

- Holds multiple pieces of information
- Information is ordered (by index)
- Can easily change what is stored in each slot
- Can store duplicate data
- Easy to iterate through

An **array** is a data structure that can hold multiple, similar values



```
String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};

int[] myNum = {10, 20, 30, 40};
```

Pros

- Holds multiple pieces of information
- Information is ordered (by index)
- Can easily change what is stored in each slot
- Can store duplicate data
- Easy to iterate through

Cons

- Can’t change the length
- Can only store one data type

Array Limitations

Cons

- **Can't change the length** *What can we do about this?*
- Can only store one data type

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

What if we wanted to add 4 to the array?

Array Limitations

Cons

- **Can't change the length** *What can we do about this?*
- Can only store one data type

```
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

Array Limitations

Cons

- **Can't change the length** *What can we do about this?*
- Can only store one data type

```
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;
```

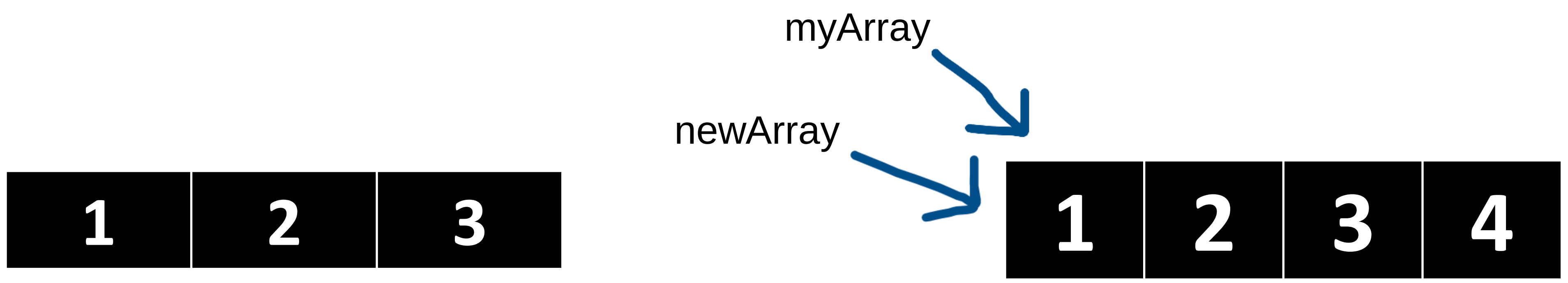
// add new value to array

// Update reference variable

Array Limitations

```java
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;
```

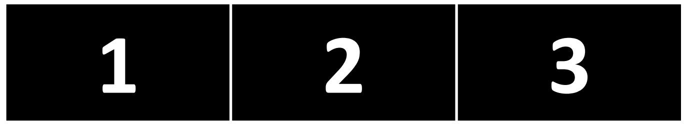
We updated our reference variable (`myArray`) to point to our new array with the new element

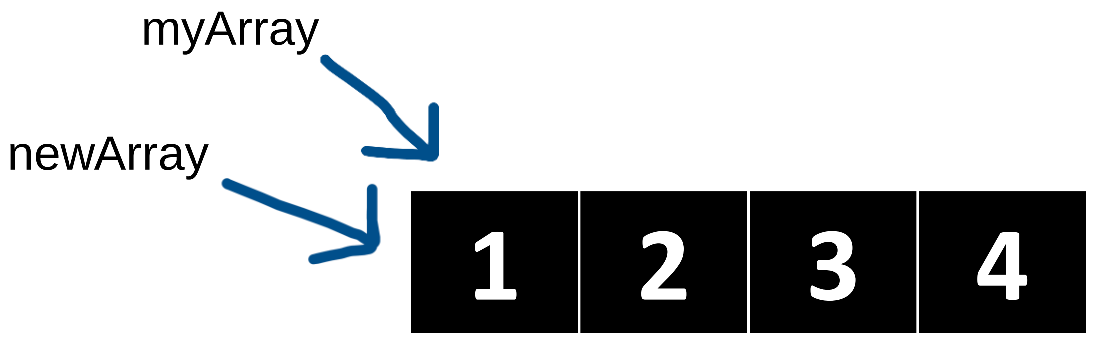


Array Limitations

```
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;
```

We updated our reference variable (`myArray`) to point to our new array with the new element





What happens to this array?
This is an unused object

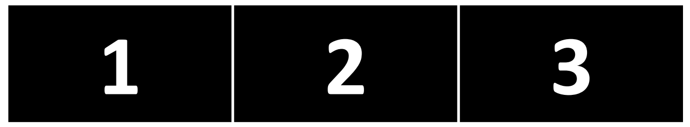
Array Limitations

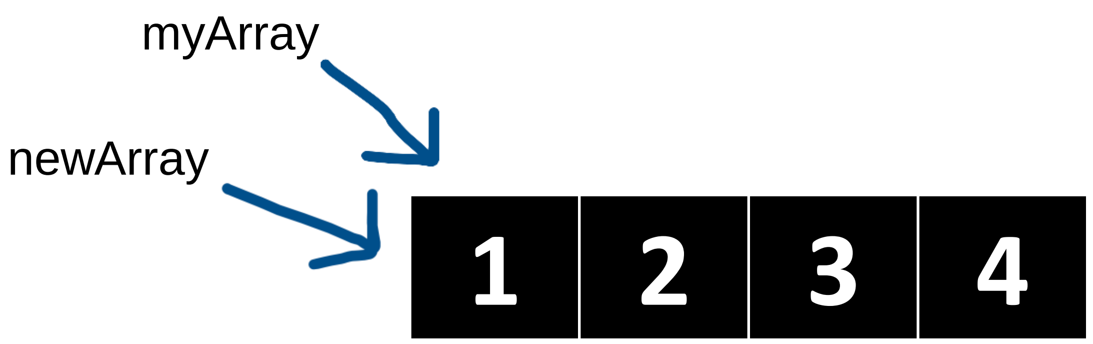
```
int[] myArray = {1, 2, 3};
System.out.println(Arrays.toString(myArray));

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;
```

We updated our reference variable (`myArray`) to point to our new array with the new element

Java has a mechanism called **Garbage Collection**, with deletes unused object to free up memory

(this runs automatically!)





What happens to this array?
This is an unused object

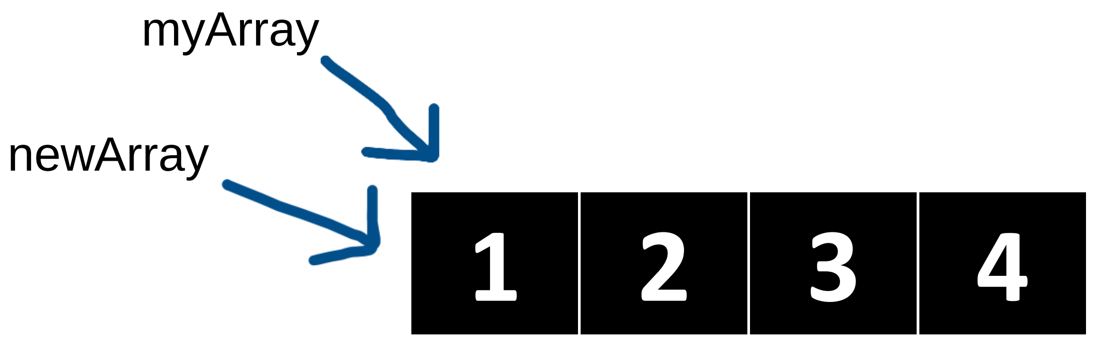
Array Limitations

```
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;
```

We updated our reference variable (`myArray`) to point to our new array with the new element

Java has a mechanism called **Garbage Collection**, with deletes unused object to free up memory

(this runs automatically!)



Java sees that we have an used/unreferenced object, so it will delete it!

Array Limitations

Cons

- **Can't change the length**

Solution

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

```java
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

We are going to write our own dynamic array data structure

Users should be able to:

1. Print the array
2. Add a new element to the array
3. Get an element at a particular index
4. Find the index of a particular element
5. Remove an element