Unit 03

Arrays

CMPS 251, Fall 2020, Dr. Abdulaziz Al-Ali

Check point

- Why do we need static methods?

- What is a very popular class that has so many static methods?

Objectives

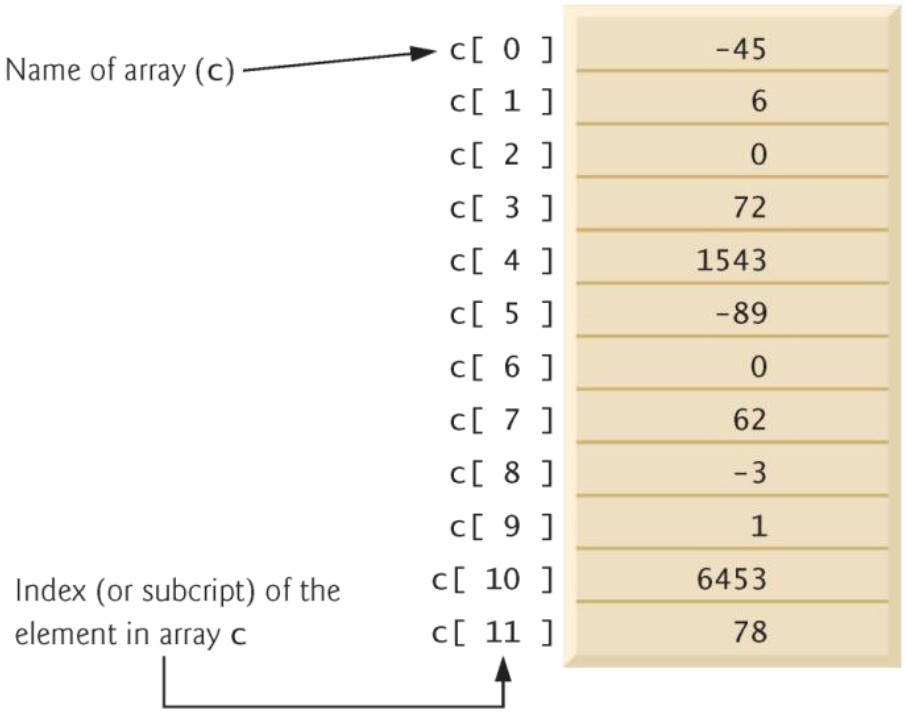
- Introduction to arrays
- Single dimensional arrays
- Arrays of objects
- ArrayList

Basic Concepts of Arrays

- A group of variables/elements of the same type
- Arrays are objects
  - Created with the **new** keyword
  - Memory allocation of an array is contiguous (elements next to each others, not randomly placed in memory)
- The array size is fixed/constant
  - Cannot be resized
  - The number of elements in the array can be retrieved using the instance variable **length**
- An array can be of any primitive or object type

Memory representation

- Example of an array of 4-byte integers



Declaring and Creating Arrays

- Basic example:

```
int[] c = new int[ 12 ]
or
int[] c; // declare the array variable
c = new int[ 12 ]; // creates the array
```

- Alternative:

```
int c[] = new int[ 12 ]
```

Arrays don't have to be of ints, floats, Strings…

- With initialized values:

```
int[] n = { 10, 20, 30, 40, 50 };
```

Check point

- Are arrays objects? Or primitive data type (like ints and floats)?
- How are arrays stored in memory?
- What happens when you create an array of Book objects?
  - More specifically, what is stored in it right after creating the array?

Simple Array Example

```
Book books[] = new Book[2];

Book b = new Book("Harry Potter");
books[0] = b;

Book c = new Book("Hunger Games");
books[1] = c;

for (int i = 0; i < books.length; i++) {
        System.out.println(books[i].getTitle());
}
```

Last lecture

- Arrays
  - Which memory are they stored in?

- When creating an array of Person objects, what is stored in position [0] right after creating the array with new Person[20];

- Fancy for loop syntax
  - for (X Y : Z)
    - What goes in each?

Demo

- See todos 1 to 7 in the code sample

Simple Array Example

```
Book books[] = new Book[2];

Book b = new Book("Harry Potter");
books[0] = b;

Book c = new Book("Hunger Games");
books[1] = c;

for (int i = 0; i < books.length; i++) {
        System.out.println(books[i].getTitle());
}
// A simpler for loop
for (Book temp : books) {
        System.out.println(temp.getTitle());
}
```

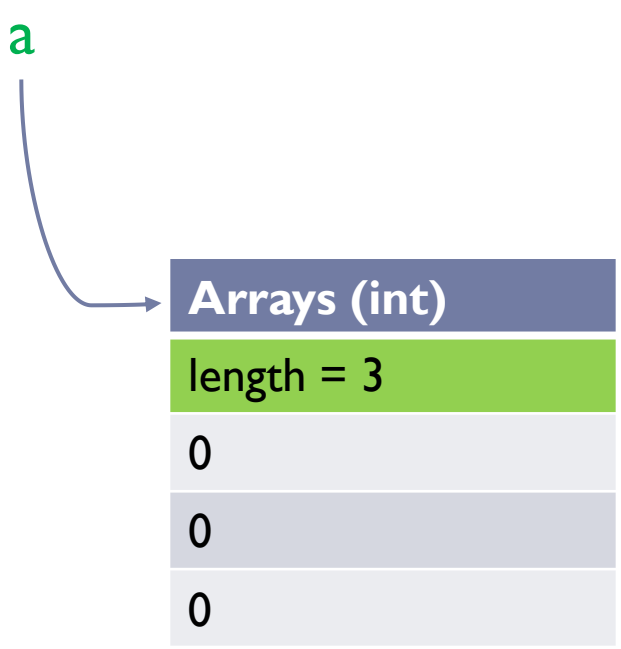
Primitives vs Objects

- Primitive variables are built in and are stored directly in memory with no object reference such as:
  - float
  - double
  - int

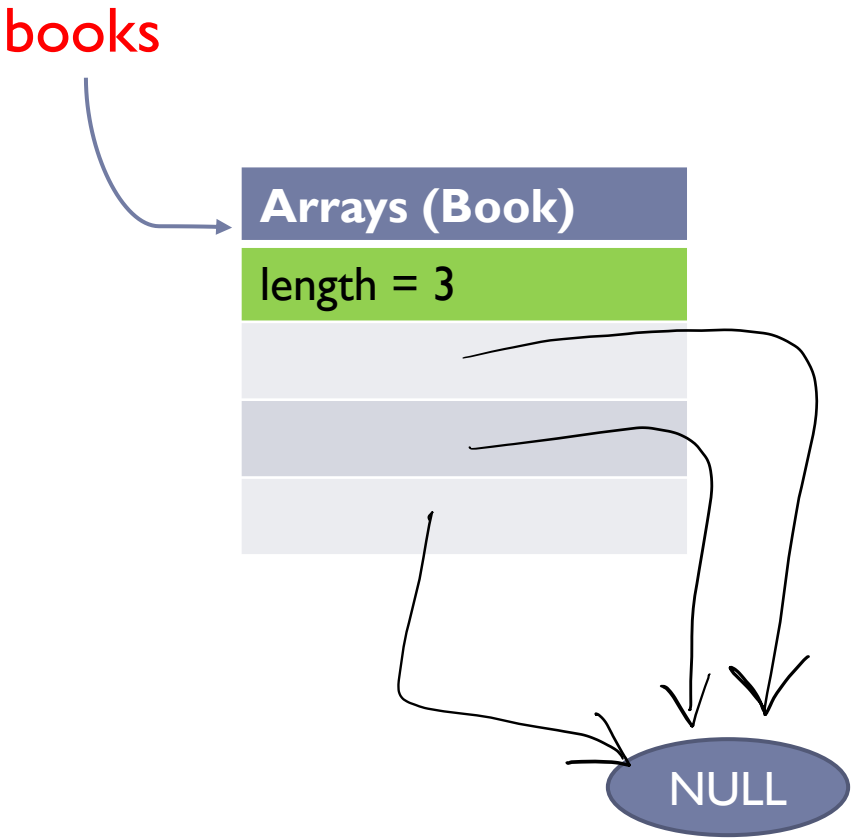
- Objects are created using the word "new" and typically by calling a constructor.
  - Book
  - String
  - Person

Array of Primitives vs Array of Objects

```
int [] a = new int[3];
```

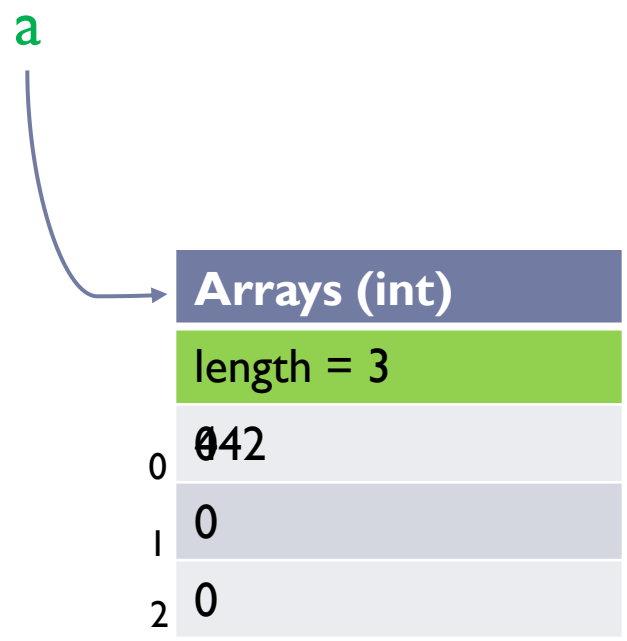



```
Book [] books = new Book[3];
```

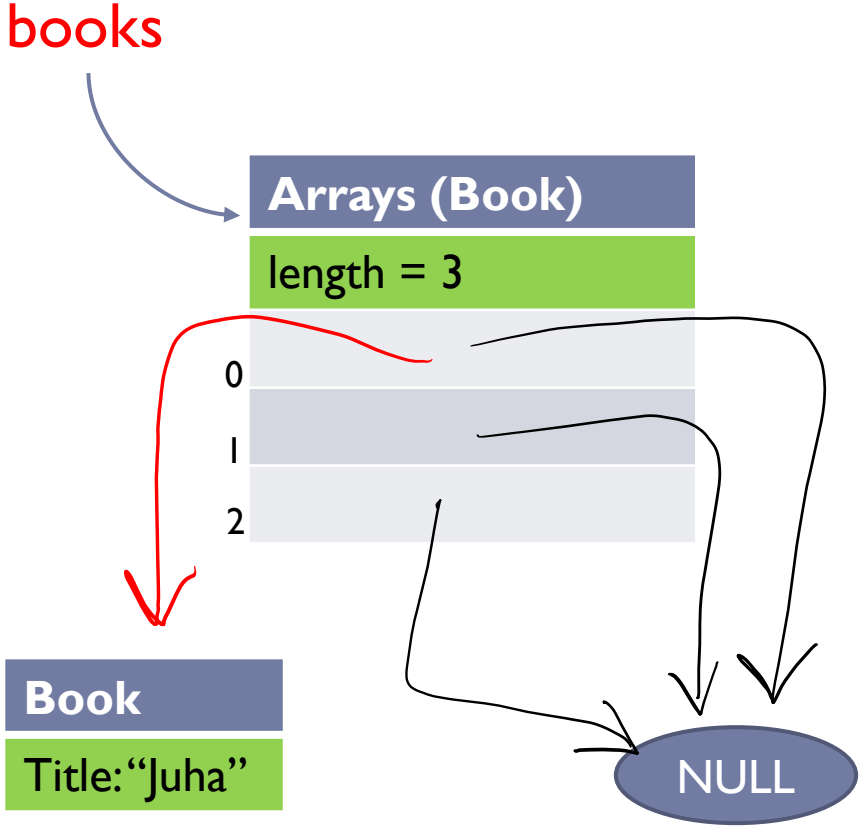


Array of Primitives vs Array of Objects

```
int [] a = new int[3];
a[0] = 442;
```




```
Book [] books = new Book[3];
books[0] = new Book("Juha")
```



Check point

- What is the difference between the following two arrays?

```
int[] intArray = new int[3];
```

```
Person[] pArray = new Person[3];
```

- Suppose only the two lines above were executed: What is stored in pArray? What about pArray[0]?

- What about intArray? And intArray[0]?

The Arrays class and its API

- **Arrays** class
  - Provides `static` methods for common array manipulations.
  - Methods include
    - sort for sorting an array (ascending order by default)
    - binarySearch for searching a sorted array
    - equals for comparing arrays
    - fill for placing values into an array.

- **System** class `static` arraycopy method.
  - Copies contents of one array into another.

Check point

- If we have an array of ints called b, how can we sort it?

- What were the parameters we specify inside arraycopy?

More about Arrays

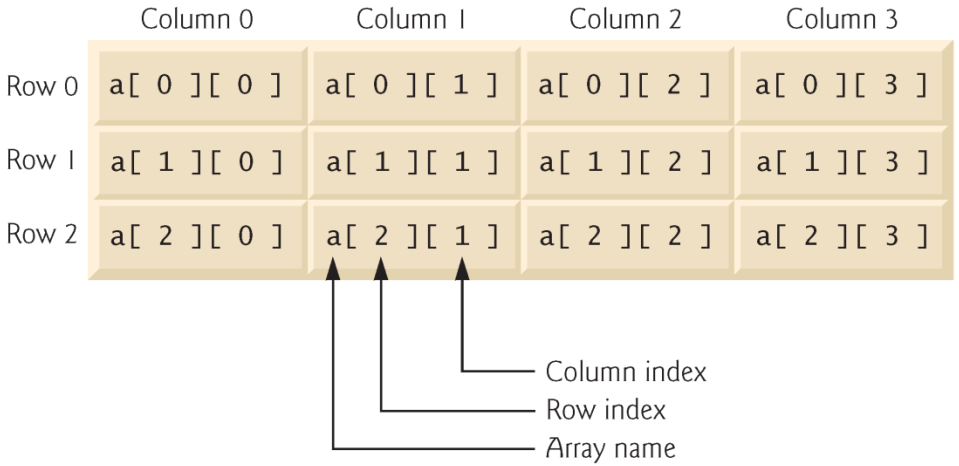
- Arrays Javadoc available from oracle here:
  https://docs.oracle.com/javase/7/docs/api/java/util/Arrays.html

Demo

- See todos 8 to 13 in the code sample

Multidimensional Arrays (EXTRA)

- Two-dimensional arrays are often used to represent tables of values.
- Multidimensional arrays can have more than two dimensions.
- Java does not support multidimensional arrays directly
  - Allows you to specify one-dimensional arrays whose elements are also one-dimensional arrays, thus achieving the same effect.
- In general, an array with *m* rows and *n* columns is called an *m*-by-*n* array.



Multidimensional Arrays Examples (EXTRA)

```
int[][] b = { { 1, 2 }, { 3, 4 } };

int[][] b = { { 1, 2 }, { 3, 4, 5 } };

int[][] b = new int[ 3 ][ 4 ];

int[][] b = new int[ 2 ][ ];   // create 2 rows
b[ 0 ] = new int[ 5 ]; // create 5 columns for row 0
b[ 1 ] = new int[ 3 ]; // create 3 columns for row 1

int[][] a;
a = new int[4][];
a[0]= new int[10];
a[1]= new int[30];
a[2]= new int[8];
a[3]= new int[15];
```

What happens if?

- We try to insert an int in position 5 and the array length is 3?

- We try to access or change the first name of a Person object in any position, right after creating the array: new Person[40]; ?

What happens if?

- We try to insert an int in position 5, but the array length is 3?
  - Program will exit with an "Exception" called ArrayIndexOutOfBoundsException.

- We try to access or change the first name of a Person object in any position, right after creating the array: new Person[40]; ?
  - Program will exit with an "Exception" called NullPointerException. Because any array of objects will start with "null" values inside of it.

Limitations of Arrays

- A standard array can't be resized
- If you want to add more elements to a full array, you can't
- What if you want to keep adding things to an array?

ArrayList

- An ArrayList is an object that stores other objects
  - Called a **collection**
- Grows dynamically, how?
  - *capacity* is increased when number of items reach the current *size*

- Check the Java API for details of how to use it:
  https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html

ArrayList Usage

```
ArrayList<Book> books = new ArrayList<Book>();

Book b = new Book("Harry Potter");
books.add(b);

Book c = new Book("Hunger Games");
books.add(c);

for(int i = 0; i < books.size(); i++) {
        Book temp = books.get(i);
        System.out.println(temp.getTitle());
}

for(Book temp: books) {
        System.out.println(temp.getTitle());
}

books.set(0, new Book("The Old Man and the Sea"); //replaces item at position 0
books.remove(1);
```

Demo

- See todos 14 to 20 in the code sample

Check point

- What are the benefits of ArrayLists?
- What are the main methods available in ArrayLists?
- How does an ArrayList know when to expand/grow?
- What are the things we used in Arrays that we cannot use in ArrayLists?
- Can we use a for-each loop for ArrayLists?

Additional Usages

- Check available static methods in Collections

  https://docs.oracle.com/javase/7/docs/api/java/util/Collections.html

Other Collections

- ArrayList
- Vector
- Stack
- LinkedList
- PriorityQueue