COMPSCI 121: ARRAYS & METHODS

SPRING 20

ARRAYS : CONTD.

- Last week: Array Operations
  - Create
  - Access
  - Modify
  - Add to

Arrays that store primitive data types such as int

- Now:
  - Arrays of Strings
  - Arrays with objects
  - Methods with Arrays
  - Two-dimensional Arrays

Arrays that store objects

Review from Last Lecture

We looked at how the data structures known as Arrays can be

- declared: using [ ]- arrays have fixed length.
- initialized: with new or specifying the elements {...}
- accessed: using [index] (indices start at 0, end length-1)
- modified/traversed: using *for* loops
- modified: using [index] and assignment.

(recall also example of expanding array capacity)

Clicker Question 1 REVIEW

Assume `j` is some `int` >= 0, and `theArray` is an array of type int and `myName` is a String.

```
1. theArray[j] = theArray[3]/2;
2. theArray[j] = theArray[(int)Math.sqrt(j)];
3. theArray[j] = theArray[j+1];
4. theArray[j] = theArray[myName.length()];
```

A. 1, 3, 4 are valid
B. 1, 2, 3, 4 are valid
C. 2, 3 are valid
D. 2, 3, and 4 are valid
E. 1, 2, 4 are valid

Clicker Question 1 REVIEW

Assume `j` is some `int` >= 0, and `theArray` is an array of type `int` and `myName` is a String.

```
1. theArray[j] = theArray[3]/2;
2. theArray[j] = theArray[(int)Math.sqrt(j)];
3. theArray[j] = theArray[j+1];
4. theArray[j] = theArray[myName.length()];
```

A. 1, 3, 4 are valid
B. 1, 2, 3, 4 are valid
C. 2, 3 are valid
D. 2, 3, and 4 are valid
E. 1, 2, 4 are valid

Clicker Question 2 REVIEW

5	10	20	30	25	50
0	1	2	3	4	5

What's wrong with this code?

```
for(int j = 0; j <= theArray.length; j++)
    System.out.println(theArray[j]);
```

A. Nothing wrong with this code
B. `j` is not incremented
C. `j` is not initialized
D. `j` takes the value of `length`

Clicker Question 2 ANSWER

5	10	20	30	25	50
0	1	2	3	4	5

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 6
```

```
for(int j = 0; j <= theArray.length; j++)
    System.out.println(theArray[j]);
```

A. Nothing wrong with this code
B. `j` is not incremented yes `j++`
C. `j` is not initialised yes `j = 0`
D. <u>`j` takes the value of `length`</u> (`length` is 6)

ARRAYS OF STRINGS

Arrays in Java are a data structure for grouping data of the same data type.

Think about: *Students* in a class; *Seats* on an airplane; *Rooms* in a motel; *Scores* on a scoreboard.

Need to keep data together as a group for processing.

Students: Bryce Jenna, Marisha, Anita, Stavros
Seats: 23A, 23B, 23C, 23D, 24A, 24B, 24C, 24D
BankAccts: 201, 300, 123, 546, 654.

INDEXING OF ARRAYS

Imagine members of a group as Objects. They can be referred to by their index numbers:

index:	[0]	[1]	[2]	[3]	[4]	[5]
Students:	Adam,	Juan,	Joey,	Tamanna,	Eduardo,	Ali

The name at index [0] is "Adam".

Note: index starts at 0!

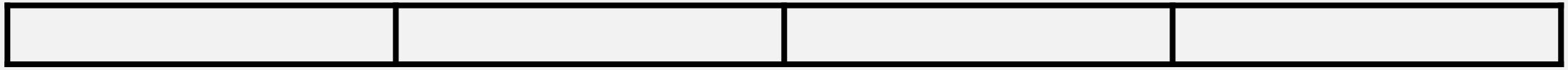
- **Code Optimization**: we can retrieve or sort the data easily.
- **Random access**: we can get any data located at any index position.

ARRAYS IN MEMORY

In your notebook or on your device, draw a memory diagram for the following declarations and number the indices:

```
int[] firstArray = new int[4];
```

e.g. boxes to draw



Now do the same for the following declaration.

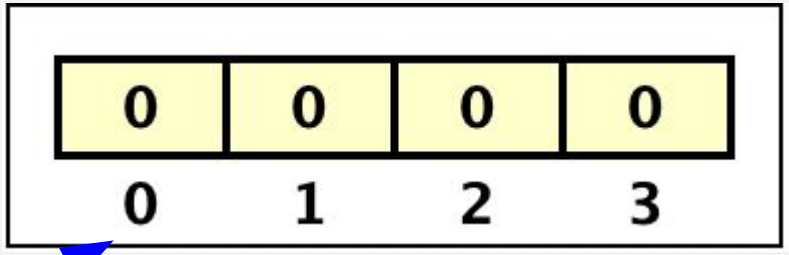
```
String[] thisIsAStringArray = new
String[4];
thisIsAStringArray[3] = "FFF";
```

STRING ARRAYS - MEMORY DIAGRAM

```
int[] firstArray = new int[4];
System.out.println(firstArray[0]);
```

Reference to the array

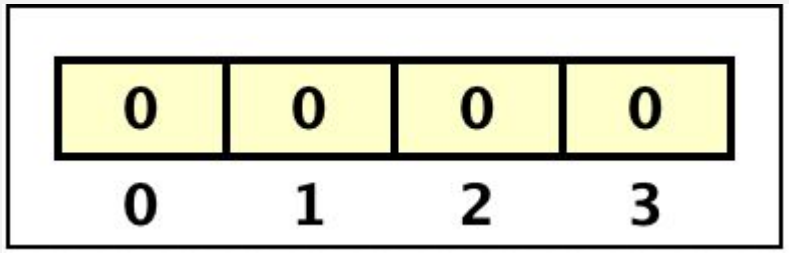
references to data in the array



There are two kinds of references- the reference to the array itself and
the references to each cell in the array, which contains a value (data).

STRING ARRAYS - MEMORY DIAGRAM

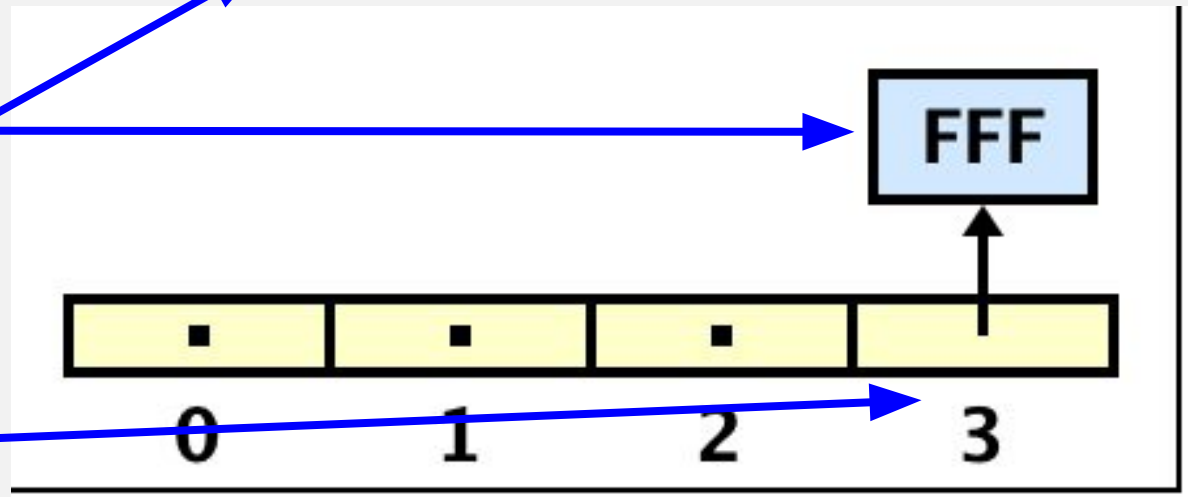
```
int[] firstArray = new int[4];
```




```
String[] thisIsAStringArray = new String[4];
thisIsAStringArray[3] = "FFF";
```

String object

Reference to the object



THINK-PAIR-SHARE

Consider the following statements:

```
String[] fruitArray = {"Apple", "Banana",
"Orange"};
fruitArray = new String[]{"Asparagus",
"Carrot"};
```

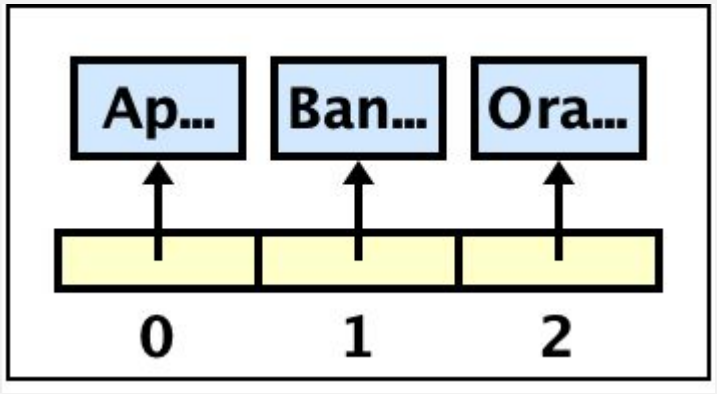
Creates a new array

What values are printed out?

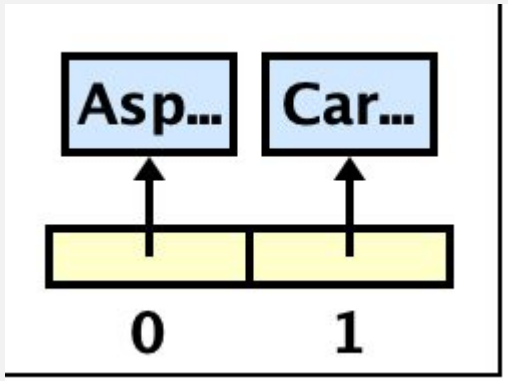
```
System.out.println(fruitArray[0]);
System.out.println(fruitArray[1]);
System.out.println(fruitArray[2]);
```

STRING ARRAYS

```
String[] fruitArray = {"Apple",
"Banana", "Orange"};
```




```
fruitArray = {"Asparagus",
"Carrot"};
```



Array reference `fruitArray` gets re-assigned!

```
System.out.println(fruitArray[0]);
System.out.println(fruitArray[1]);
System.out.println(fruitArray[2]);
```

```
Asparagus
Carrot
java.lang.ArrayIndexOutOfBoundsException: array index (2) is greater than or equal to array size (2)
```

Clicker Question 3

```
String[] arr = new String[5];
String dataStr1 ="salt & vinegar crisp, ";
arr[0] = " egg,";
arr[1] = " potato,";
arr[2] = " cream cheese, ";
arr[3] = dataStr1;
for(int i=0;i<arr.length;i++)
 System.out.print(arr[i]);
```

What is printed?

A. salt & vinegar crisp, egg, potato, cream cheese
B. egg, potato, cream cheese, salt & vinegar crisp, null
C. egg, potato, cream cheese, salt & vinegar crisp
D. Does not compile
E. egg, potato, cream cheese, null

Clicker Question 3 ANSWER

```
String[] arr = new String[5];
String dataStr1 ="salt & vinegar crisp, ";
arr[0] = " egg,";
arr[1] = " potato,";
arr[2] = " cream cheese, ";
arr[3] = dataStr1;
for(int i=0;i<arr.length;i++)
 System.out.print(arr[i]);
```

```
arr[4] is
null
```

A. salt & vinegar crisp, egg, potato, cream cheese
B. <u>egg, potato, cream cheese, salt & vinegar crisp, null</u>
C. egg, potato, cream cheese, salt & vinegar crisp
D. Does not compile
E. egg, potato, cream cheese, null

ARRAY OF OBJECTS

```
public Fruit(String t, String c, int age, boolean seeds) {
        type = t;
        color = c;
        daysOld = age;
        hasSeeds = seeds;}
```

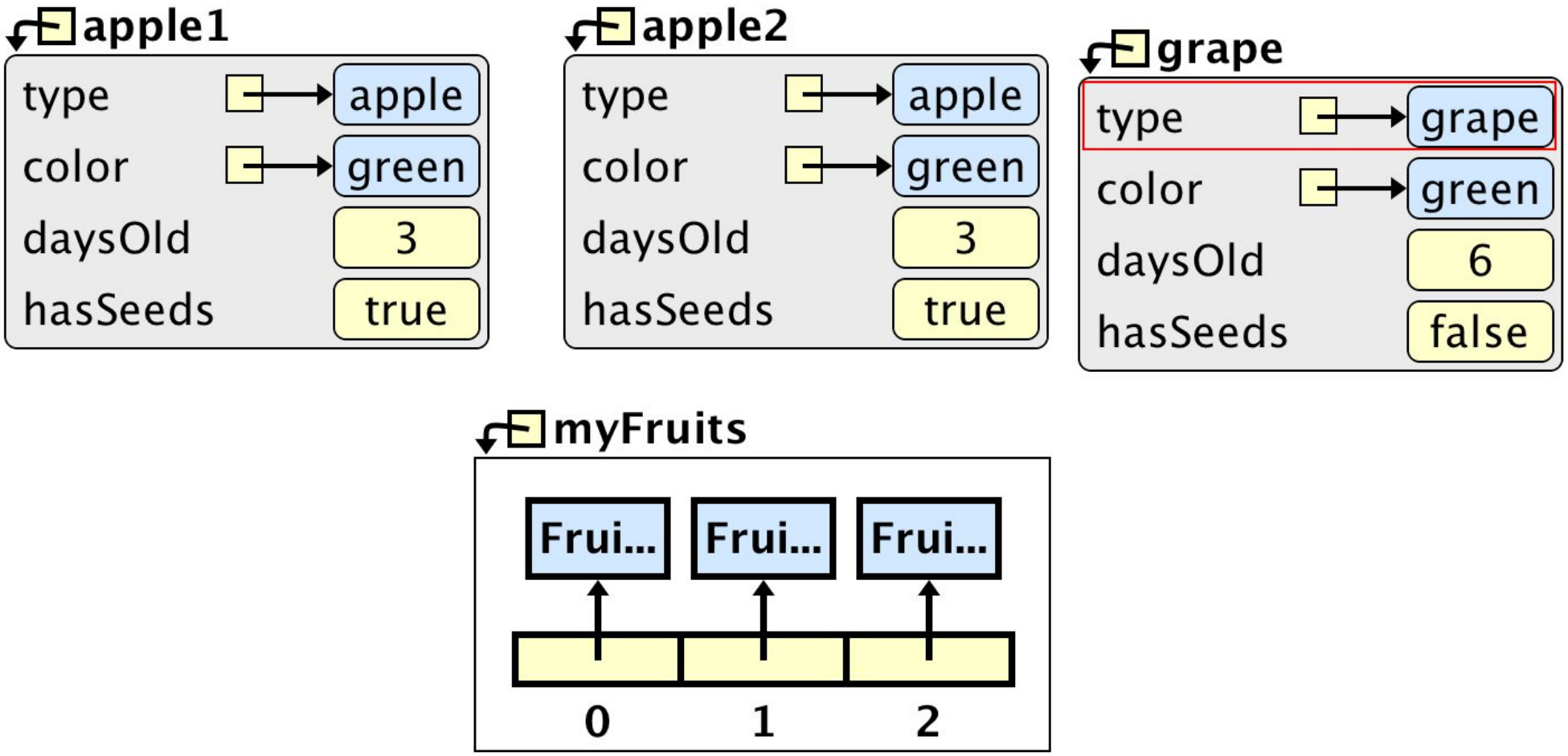
Constructor in the `Fruit` class

Create Fruit objects and an array to store them (in a main method).

```
Fruit apple = new Fruit("apple", "green", 3, true);
Fruit apple2 = new Fruit("apple", "green", 3, true);
Fruit grape = new Fruit("grape", "green", 6, false);
Fruit[] myFruits = {apple, grape, apple2};
Fruit.printFruitArray(myFruits);
```

use objects to initialize array

ARRAY OF `FRUIT` OBJECTS



DEMO: `Fruit.java`

ARRAY AS METHOD PARAMETER

Array as parameter

```
public static void printFruitArray(Fruit[] fruits)
{
  for (Fruit f : fruits) {
    String message = f.getDaysOld() + "-day-old ";
        if (!f.hasSeeds()) {
          message += "un";}
        message += "seeded " + f.getColor() + " " +
f.getType();
        System.out.println(message);}}
```

Call method on object

Enhanced *for* loop-

TRAVERSING ARRAY WITH ENHANCED `FOREACH` LOOP

```
public static void printFruitArray(Fruit[] fruits)
```

foreach loops that apply to arrays march down an entire array of objects, either:

- Collecting information; or
- Altering the contents of objects

```
for (Fruit f : fruits)
```

Type tag variable colon array-name

PROBLEMS WiTH USING `FOREACH` LOOP

You cannot:

- change the array in any way.
- traverse or compare two arrays.
- iterate backward, only forward by single steps.

Clicker Question 4

Assume an `Infant` class

```
public String oldest(Infant[] kids){
 if (kids.length == 0) return( "no kids");
 Infant oldKid = kids[0];
  foreach statement ??? {
   if (k.getAge() > oldKid.getAge()) oldKid = k;
  }
 return(oldKid.getName());
}
```

The missing statement is ?

A. `for(Infant k : oldKid)`
B. `for(Infant k : kids)`
C. `for(Infant k : kids);`
D. `for(Infant k ; kids)`
E. `for(Infant[] k : kids)`

Clicker Question 4 ANSWER

```
public String oldest(Infant[] kids){
 if (kids.length == 0) return( "no kids");
 Infant oldKid = kids[0];
  foreach statement here {
   if (k.getAge() > oldKid.getAge()) oldKid = k;
  }
 return(oldKid.getName());
}
```

The missing statement is ?

A. `for(Infant k : oldKid)` //wrong array name
B. `for(Infant k : kids)`
C. `for(Infant k : kids);` //semicolon error
D. `for(Infant k ; kids)` //missing colon
E. `for(Infant[] k : kids)` //[] brackets error

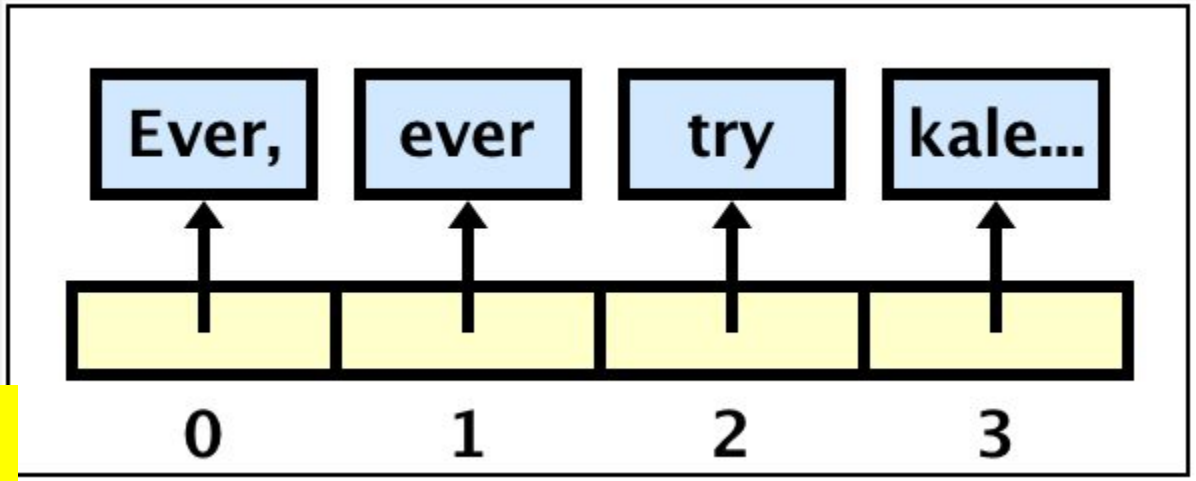
CREATING AN ARRAY FROM STRINGS

```
String s = "Ever, ever try kale??";
String[] words = s.split(" ");
for (String st : words){
  System.out.println (st);
}
```

Ever, ever try kale??

```
Ever,
ever
try
kale??
```

```
words --> (obj 156 : java.lang.String[4]) java.lang.St
  [0] --> "Ever," (obj 157 : java.lang.String) java.lan
  [1] --> "ever" (obj 159 : java.lang.String) java.lang
  [2] --> "try" (obj 160 : java.lang.String) java.lang.
  [3] --> "kale??" (obj 161 : java.lang.String) java.la
```



See: `TokenTest.java`

RETURNING AN ARRAY FROM METHODS

Returns an array

```
public static int[] arrayCreator() {
        int[] anArray = {1,2,3,4,5};
            return anArray;
}
```

Construct array

return statement

Clicker Question 5

```
public static int[] initializeArray(int value, int size) {
      int[] resultArray = new int[size];
      int index;
      for (index = 0; index < resultArray.length; ++index){
         resultArray[index] = value;
       }
        return resultArray;
}
```

A. null
B. {}
C. {5,5,5,5,5,0,0}
D. {5,5,5,5,5,5,5}
E. {7,7,7,7,7}

What does the returned array contain after this method call?

`initializeArray(5, 7);`

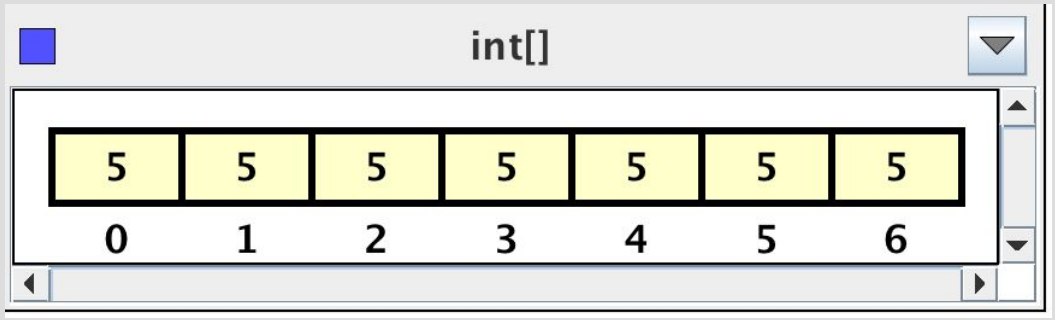
Clicker Question 5 ANSWER

```
public static int[] initializeArray(int value, int size) {
      int[] resultArray = new int[size];
      int index;
      for (index = 0; index < resultArray.length; ++index) {
         resultArray[index] = value;}
        return resultArray;}
```

What does the returned array contain after this method call?

`initializeArray(5, 7);`

A. null
B. {}
C. {5,5,5,5,5,0,0}
D. {5,5,5,5,5,5,5}
E. {7,7,7,7,7}



Problem Statement 1

Every week, I create an array for my shopping list for 5 items.

However, when I have less than 5 items, it prints the following:

```
Milk
Oranges
Apples
null
null
```

Problem of the oversized array!

Problem Statement 1a

```java
public class ShoppingList{

  public static void main(String[] args)
  {

   // Construct an empty list with 5 elements
   String[] shoppingList = new String[5];
   shoppingList[0] = "Milk";
   shoppingList[1] = "Oranges";
   shoppingList[2] = "Apples";

  //print my weekly shopping items
   for (int index = 0; index < shoppingList.length; index++) {
      System.out.println(shoppingList[index]);
   }

  }//end main

}//end class
```

Here is the code I am using.

How do I stop my program from printing null?

Possible Solution for Problem Statement 1c

```
public class ShoppingListRevised{

  public static void main(String[] args)
  {

    // Construct an empty list with 5 elements
    String[] shoppingList = new String[5];
    int shoppingListSize = 0;

    // Add first element to shopping list
    shoppingList[shoppingListSize] = "Milk";
    shoppingListSize++;

    // Add second element to shopping list
    shoppingList[shoppingListSize] = "Oranges";
    shoppingListSize++;

    // Add third element to shopping list
    shoppingList[shoppingListSize] = "Apples";
    shoppingListSize++;
```

Solution: Use a separate integer variable to keep track of how many array elements are currently used and keep incrementing it.

Possible Solution for Problem Statement 1d

```
/*
for (int index = 0; index < shoppingList.length; index++) {
    System.out.println(shoppingList[index]);
}
*/
for (int index = 0; index < shoppingListSize; index++) {
    System.out.println(shoppingList[index]);
}
```

```
Milk
Oranges
Apples
```

Keep track of:

- the array reference **variable name of array**
- the current size **an integer variable**
- array capacity to check if array is full
  `arrayName.length`

Clicker Question 6

How do I swap items on my list? E.g. Apples and Milk change position. Will this code work?

```
shoppingList[0] = shoppingList[2];
shoppingList[2] = shoppingList[0];
```

```
Milk
Oranges
Apples
```

A. Compiler error
B. Yes, it works and the list is now: Apples, Oranges, Milk
C. No, it does not work and the list is now: Apples, Oranges, Apples
D. No, it does not work and the list is now: Oranges, Apples, Milk

Clicker Question 6 Answer

```
shoppingList[0] = shoppingList[2];
shoppingList[2] = shoppingList[0];
```

```
shoppingList --> (obj 167 : java.lang.String[5])  java.lang.String[]
  [0] --> "Apples" (obj 175 : java.lang.String)  java.lang.String
  [1] --> "Oranges" (obj 174 : java.lang.String)  java.lang.String
  [2] --> "Apples" (obj 175 : java.lang.String)  java.lang.String
```

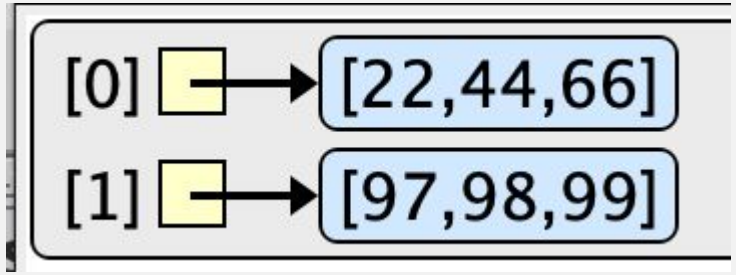
A. Compiler error
B. Yes, it works and the list is now: Apples, Oranges, Milk
C. No, it does not work and the list is now: Apples, Oranges, Apples
D. No, it does not work and the list is now: Oranges, Apples, Milk

TWO DIMENSIONAL ARRAYS

```
// Initializing a 2D array
int[][] numVals = { {22, 44, 66}, {97, 98, 99} };

// Use multiple lines to make rows more visible
int[][] numVals = {
        {22, 44, 66}, // Row 0
        {97, 98, 99}  // Row 1
};
```

	0	1	2
0	22	44	66
1	97	98	99



USEFUL LINKS

Read more about arrays at:

https://docs.oracle.com/javase/specs/jls/se8/html/jls-10.html

https://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html

Links to to practice exercises:

https://codingbat.com/java/Array-1

https://introcs.cs.princeton.edu/java/14array/

TO-DO

- Check your iClicker grades in Moodle.
- Complete zyBook chapter 7 exercises.
- Communicate with us using only Moodle forum or Piazza.
- Submit Project 3 early and *often* - seek help in office hours.
- Be ready for Exam 2.