1.00 Lecture 11

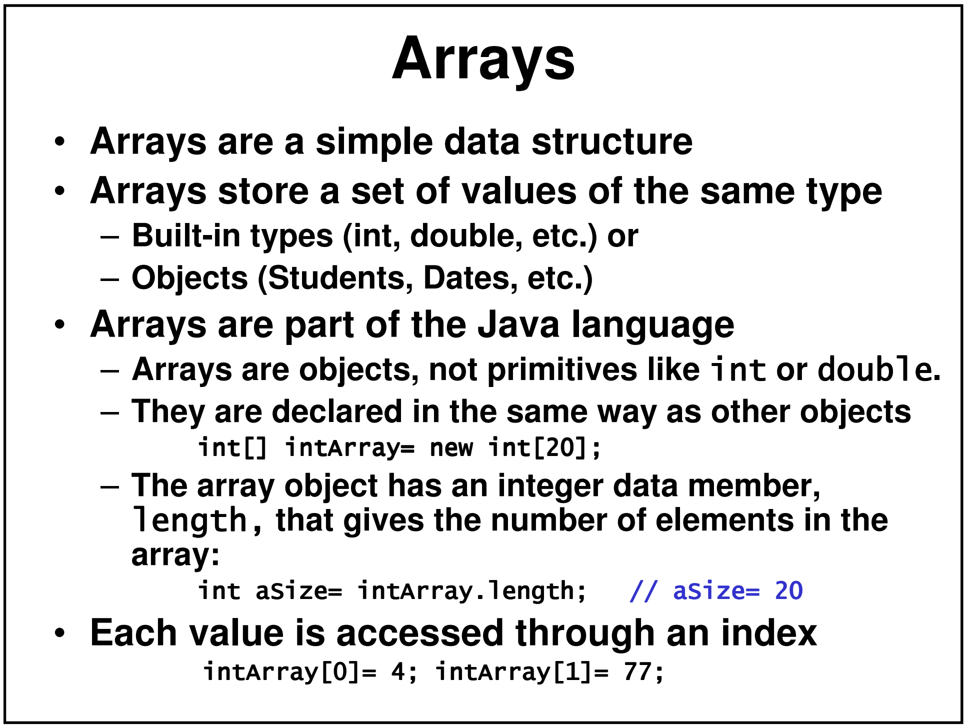
Arrays and ArrayLists

Reading for next time: Big Java: sections 17.1, 17.3, 17.5

Strings

```
public class StringExample {
  public static void main(String[] args) {
    String s= new String("Test");      // Strings are objects
    String first= "George ";           // Shortcut constructor
    String middle= "H.W. ";
    String last= "Bush";
    String full= first + middle + last;
    System.out.println("Full: " + full);

    // Testing for equality in strings (objects in general)
    String full2= "George H.W. Bush";
    if (full.equals(full2))      // Right way
        System.out.println("Strings equal");
    if (full == full2)           // Wrong way
        System.out.println("A miracle!");
    if (first == "George ")  // Wrong way,but sometimes works
        System.out.println("Not a miracle!");
    // Modifying strings must be done indirectly-constant
    middle= middle.substring(2, 4) + " ";
    full= first + middle + last;
    System.out.println("Modified full: " + full);    }  }
```



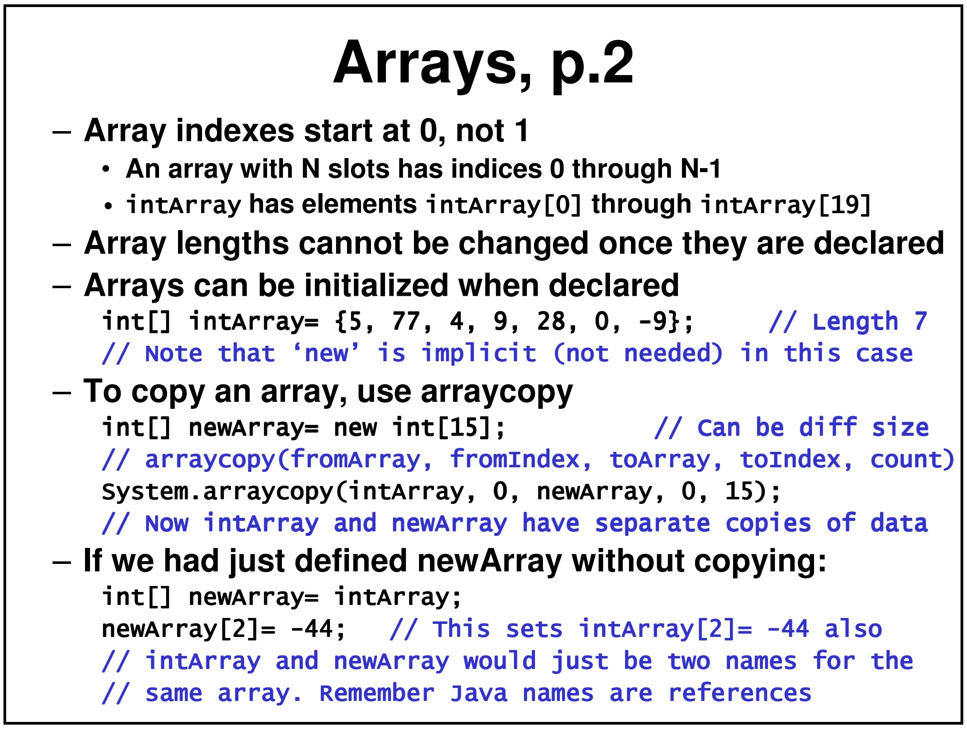
Arrays

- Arrays are a simple data structure
- Arrays store a set of values of the same type
  - Built-in types (int, double, etc.) or
  - Objects (Students, Dates, etc.)
- Arrays are part of the Java language
  - Arrays are objects, not primitives like `int` or `double`.
  - They are declared in the same way as other objects

    ```
    int[] intArray= new int[20];
    ```
  - The array object has an integer data member, `length`, that gives the number of elements in the array:

    ```
    int aSize= intArray.length;   // aSize= 20
    ```
- Each value is accessed through an index

  ```
  intArray[0]= 4; intArray[1]= 77;
  ```



Arrays, p.2

- Array indexes start at 0, not 1
  - An array with N slots has indices 0 through N-1
  - `intArray` has elements `intArray[0]` through `intArray[19]`
- Array lengths cannot be changed once they are declared
- Arrays can be initialized when declared

  ```
  int[] intArray= {5, 77, 4, 9, 28, 0, -9};     // Length 7
  // Note that ‘new’ is implicit (not needed) in this case
  ```
- To copy an array, use arraycopy

  ```
  int[] newArray= new int[15];          // Can be diff size
  // arraycopy(fromArray, fromIndex, toArray, toIndex, count)
  System.arraycopy(intArray, 0, newArray, 0, 15);
  // Now intArray and newArray have separate copies of data
  ```
- If we had just defined newArray without copying:

  ```
  int[] newArray= intArray;
  newArray[2]= -44;   // This sets intArray[2]= -44 also
  // intArray and newArray would just be two names for the
  // same array. Remember Java names are references
  ```

Test Your Knowledge

1. Which of the following expressions does not declare and construct an array?
   a. int[ ] arr = new int[4];
   b. int[ ] arr;
   arr = new int [4];
   c. int[ ] arr = {1,2,3,4};
   d. int[ ] arr;

2. Given this code fragment:

```
int j= ?;
int[] data = new int[10];
System.out.print(data[ j ]);
```

Which of the following is a legal value of j?
a. -1
b. 0
c. 1.5
d. 10

Test Your Knowledge

3. Given this code fragment:

```
 int[] arrayA = new int[4];
int[] arrayB;
arrayB = arrayA;
arrayB[2]=4;
arrayA[0]=arrayB[2];
```

What are the values of the elements in array A?
a. unknown
b. 0,0,0,0
c. 4,0,4,0
d. 4,0,0,0

4. How many objects are present after the following code fragment has executed?

```
double[] arrayA=new double[10];
double[] arrayB;
arrayB = arrayA;
```

a. 1
b. 2
c. 10
d. 20

Test Your Knowledge

5. For which of these applications an array is NOT suitable?

- **a. Holding the scores on 4 quarters of a Basketball game**
- **b. Holding the name, account balance and account number of an individual**
- **c. Holding temperature readings taken every hour through a day**
- **d. Holding monthly expenses through a year**

6. Given the following code fragment:

```
int[] data = {1,3,5,7,11};
for(______________________)
System.out.println(data[index]
    );
```

Fill in the blanks so that the program prints out every element in the array in order

- **a. int index = 4; index>0; index--**
- **b. int index=0; index<4; index++**
- **c. int index=0; index<data.length(); index++**
- **d. int index=0; index<data.length; index++**

Test Your Knowledge

7. What is the output of the following program?

```
public class Test{
  public static void main ( String[] args ){
    int value = 10;
    int[] arr = {10,11,12,13};
    System.out.println("value before:"+value);
    alterValue( value );
    System.out.println("value after:"+value);
    System.out.println("arr[0]before:"+arr[0]);
    alterArray( arr );
    System.out.println("arr[0] after:"+arr[0]);
  }
  public static void alterValue (int x ){
    x = 0; }
  public static void alterArray (int[] a){
    a[0] = 0; }
}
```

a. value before:10
value after:0
arr[0] before:10
arr[0] after: 0

b. value before:10
value after:10
arr[0] before:10
arr[0] after: 10

c. value before:10
value after:10
arr[0] before:10
arr[0] after: 0

d. value before:10
value after:0
arr[0] before:10
arr[0] after: 10

Exercise

- A. Create a TemperatureTest class
- B. Start writing main():
  - Declare and construct an array of doubles, called `dailyTemp` holding daily temperature data
    - Use an initializer list with braces { }

Mon	Tue	Wed	Thu	Fri	Sat	Sun
70	61	64	71	66	68	62

  - Using a for loop, print every element of the `dailyTemp` array in reverse order (starting from Sunday and going backwards to Monday)
    - Use the array length, not the constant 7, to control the loop

Exercise, p.2

- C. Create a Temperature class. In it, write a method to find average weekly temperature:

```
public static double average(double[] a) {
   // Code goes here
   // Declare a total variable, initialize it to 0
  // Loop thru a and add each element to the total
  // Divide by the number of elements, return the answer
}
```

- D. In the TemperatureTest main() method, call the average method you just wrote
  - Call it as Temperature.average(…);
  - Pass the dailyTemp array as the argument
  - Average method returns a double that you should store in variable averageTemp in main()
  - Print the average temperature in main() as:
    - Average weekly temperature: 66

ArrayList

- ArrayList class is a fancy version of an array
  - ArrayList can grow automatically as needed
    - Has `capacity` that is increased when needed
    - Has `size`, which is the actual number of elements
  - ArrayList can hold elements of different types
    - As long as each is an Object (reference),
    - ArrayList can't hold a basic type (int, double, etc.) !
    - You will use type casts or wrapper classes in ArrayList
      - Wrappers are objects (e.g., Double) that hold built-ins
  - ArrayList is not in the core language
    - It is in package java.util, which you must import:

```
import java.util.*;      // At top of program
```

  - ArrayLists are slightly slower than arrays
    - It matters in large scale numerical methods
  - ArrayLists have many methods you can use

ArrayList Methods

`void add(Object o)`	Add object to end of ArrayList, increases ArrayList size by one
`void add(int i, Object o)`	Insert object at index i
`Object get(int i)`	Return object at index i. Usually cast to actual type `Car c= (Car) a.get(4);`
`int indexOf(Object o)`	Find first occurrence of object; uses equals method
`boolean isEmpty()`	Return true if ArrayList has no objects, false otherwise
`void remove(int i)`	Delete object at index i
`void set(int i, Object o)`	Replace element at index i with specified object
`int size()`	Return number of Objects in ArrayList

To create new ArrayList: ArrayList a= new ArrayList();

Test Your Knowledge

1. Which of the following statements is NOT true about ArrayLists?

   a. ArrayLists are slightly faster than arrays.

   b. ArrayLists can store elements of different types.

   c. ArrayLists can increase in size to store more elements.

   d. ArrayLists have methods to manage their content.

2. Considering myArrayList is a ArrayList that has been declared and constructed, which of the following statements is always true?

   a. myArrayList.size() >= myArrayList’s capacity

   b. myArrayList.size() > myArrayList’s capacity

   c. myArrayList’s capacity >= myArrayList.size()

   d. myArrayList’s capacity > myArrayList.size()

Test Your Knowledge

3. Given the following code fragment:

```
ArrayList myArrayList = new ArrayList();
myArrayList.add("One");
myArrayList.add("Two");
myArrayList.add("Three");
myArrayList.add("Four");
```

Which one of the following expressions will modify myArrayList so it looks like:

One; Two; Four

a. myArrayList.remove(myArrayList.get(3));

b. myArrayList.remove(myArrayList.indexOf("Three"));

c. myArrayList.remove(3);

d. myArrayList.remove(myArrayList.get("Two"));

Test Your Knowledge

4. Given the code fragment of question 3, which one of the following expressions will modify myArrayList so it looks like:

   One; Two; Three; Five

   a. myArrayList[3] = "Five"
   b. myArrayList[4] = "Five"
   c. myArrayList.set(myArrayList.indexOf("Four"), “Five”);
   d. myArrayList.set(myArrayList.indexOf("Five"), “Four”);

Test Your Knowledge

5. Given the code fragment of question 4, which one of the following expressions will modify myArrayList so it looks like:

   One; Two; Three

   a. myArrayList.remove(2);
   b. myArrayList.remove(myArrayList.lastElement());
   c. myArrayList.remove(myArrayList.size());
   d. myArrayList.remove(myArrayList.size()-1);

Test Your Knowledge

6. Given the following code fragment:

```
ArrayList myArrayList = new ArrayList();
myArrayList.add(new Integer(1));
myArrayList.add(new Integer(3));
myArrayList.add(new Integer(7));
```

Which one of the following expressions will modify myArrayList so it looks like:

1 3 5 7

a. myArrayList.add(new Integer(5));

b. myArrayList.add(2, new Integer(5));

c. myArrayList.add(2, 5);

d. myArrayList.add(3, 5);

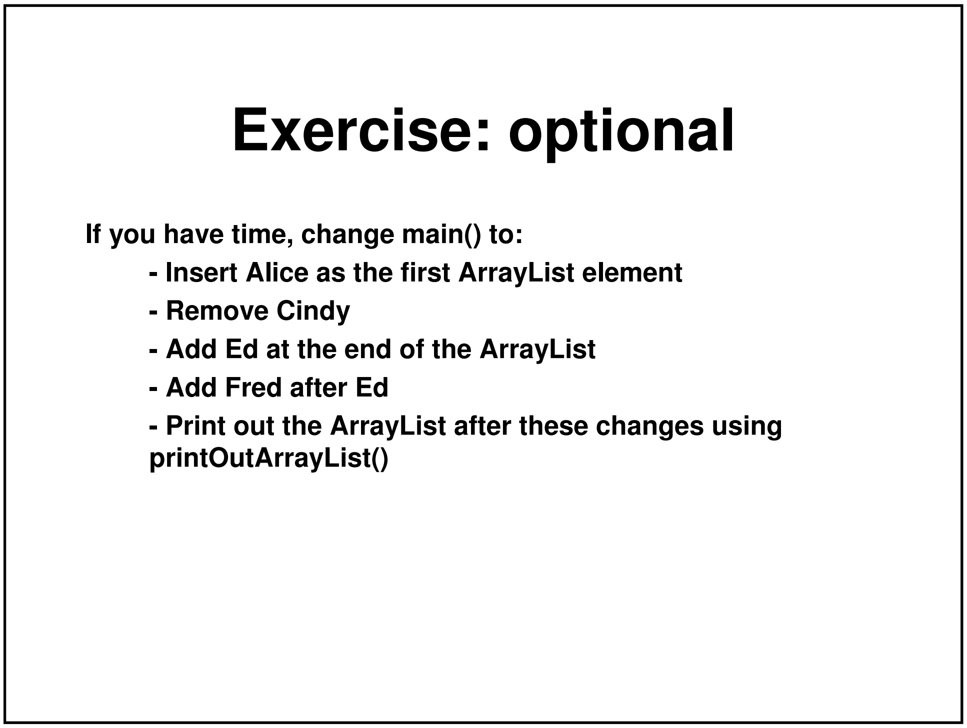
Exercise

- Exercise: Store students in a course
  1. Create a class MITCourseTest.
  2. In its main() method:
     - Create a ArrayList `students`
     - Add 4 students to the ArrayList:
       - “Amy”, “Bob”, “Cindy” and “David”
       - Add them to the ArrayList directly:

         ```
         students.add("Amy");
         ```
  - Create a class MITCourse:
    - Write a method to print all the elements in the ArrayList and its size

    ```
    public static void printOutArrayList(ArrayList arr) {
            // Code goes here
    }
    ```
  - Call printOutArrayList() method from main().
    - Pass the ArrayList as the argument
  - Your output should be:

    ```
    Amy
    Bob
    Cindy
    David
    Size: 4
    ```



Exercise: optional

If you have time, change main() to:

- Insert Alice as the first ArrayList element
- Remove Cindy
- Add Ed at the end of the ArrayList
- Add Fred after Ed
- Print out the ArrayList after these changes using printOutArrayList()