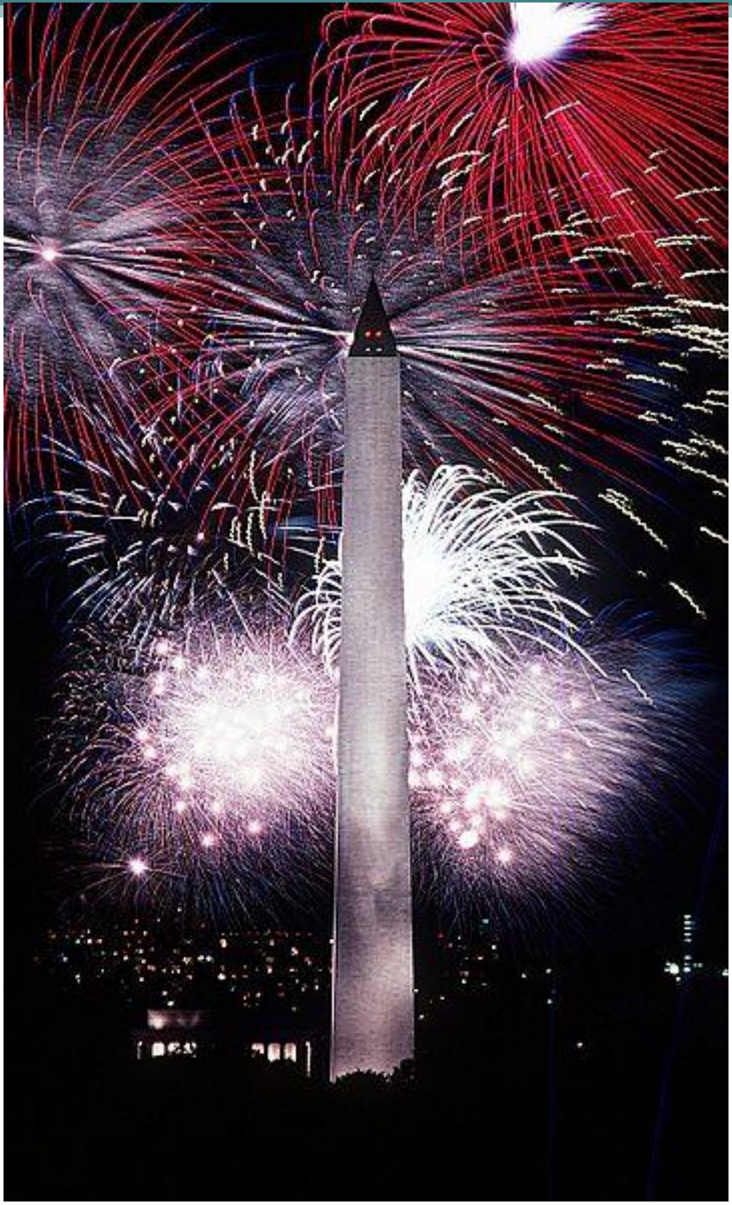
CS 302: Introduction to Programming in Java

Lecture 9

THE UNIVERSITY of WISCONSIN MADISON



No class on Wednesday

in Observance of
Fourth of July

THE UNIVERSITY of WISCONSIN MADISON

Announcement

- Programming Assignment #1 Due 11:59pm Sunday July 8th
- Follow style and commenting guide (20% of total grade)
- Let me know by Independence Day if you'd like to work with a partner

Schedule

- This week:
  - Loop Review
  - Arrays
    - 6.1-6.3
  - Methods
    - 5.1-5.2
  - Arrays with Methods and 2D Arrays
    - 5.2-5.7
    - 6.4-6.5

THE UNIVERSITY of WISCONSIN MADISON

Review

- What are the three types of loops? When should each be used?
- When does infinite loop happen?
- break && continue statements

Review

- What is the output of the following statement:

```
for (int i = 1; i < 4; i++)
{
   int j = 1;
   while(j <= i) {
       S.o.pln(i*j);
       j++;
   }
}
```

THE UNIVERSITY *of* WISCONSIN MADISON

Review

- What is the value of x at the end of the following statement:

```
int x = -1;
if (x < 4) {
  S.o.pln("x^2 = " + Math.pow(x, 2));
  x++;
}
S.o.pln(x);
```

A) 1
B) 4
C) 0
D) 5
E) None of the above

Practice 1

Write a program to assist an instructor in calculating grades for a single student

Input: multiple scores (each out of 100), -1 to quit

- Example input: 77(then hit enter) 90(then hit enter) 57(…) 88(…) -1(…)

Output:

- The number of scores entered
- The sum of all scores entered
- Highest single score
- Average score

Don't count the -1 as a valid score!

1). Validate input type (must be int); 2). Scores must be 0<= scores <= 100 which should be checked by the program!!

Array Basics

- A means of collecting and storing a bunch of values with the same type
- Defined with brakets - []
- Arrays have a type and name just like primitive variables
- Arrays have a fixed size
- Analogy = bookshelf, carriages in a train, mailboxes at USPS

Array Example

What if we had a bunch of test scores:

99, 95.6, 78, 82.9, 85.2, 88

Could just have user enter them and store each one in its own variable, however using an array is a better choice

Must match array type

```
double[] testScores = new double[6];
```

Array type

Brackets indicate array

Array variable name

Size of the array

Declaration and Initialization

- Declaration
  - double[] testScores;
  - String[] stringArray;
- Initialization
  - If we don't immediately know what will go in the array (all elements set to 0 (if int/double) false (if boolean) and null (if String))
    - testScores = new double[6];
    - words = new String[10];
  - If we know what we want in the array
    - testScores = {99, 95.6, 78, 82.9, 85.2, 88};
    - stringArray = {"a String", "another String", "etc."};

<u>Note in either case we need to know the size of the array at Initialization!</u>

Indexing and Accessing Elements

double[] testScores = {99, 95.6, 78, 82.9, 85.2, 88};

- 6 elements so length of testScores is 6
  - Stored in: testScores.length; //notice no () after length

testScores

- 0-indexed like Strings
- int x = testScores.length; //x now equals 6
- double y = testScores[0]; //y now equals 99
- double z = testScores[x]; //index out of bounds exception – the last //value is stored in index 5

THE UNIVERSITY of WISCONSIN MADISON

Modifying Array Contents

- Remember to stay within the bounds
  - 0 <= index < array.length
- Modifying array contents is as easy as accessing the index and setting it to a new value

```
double[] testScores = {99, 95.6, 78, 82.9, 85.2, 88};



testScores[3] = 2; //0 <= 3 < 6 so we are within the bounds
```

- Think of each index in the array as a separate variable of type int

For Loop

- Often we use the for loop to iterate through arrays (why?)

Ex. find the average test score:

```
double[] testScores = {99, 95.6, 78, 82.9, 85.2, 88};
double sum = 0;
for (int i = 0; i < testScores.length; i++) //why do we start i at 0?
{
  sum += testScores[i];
}
double avg = sum / testScores.length;
```

String as parameter in Scanner

- String str = “1 2 3 4 5 6 7 8 9 10”
- Scanner in = new Scanner(str);
- in.hasNextInt() always true; in.nextInt() can traverse all 10 integer numbers in the str input stream
- Example:

```
1.  String str = “1 Madison 2 3 4 Wisconsin 6”;
2.  Scanner in = new Scanner(str);
3.  while (in.hasNext()) {
                    if (in.hasNextInt())
                              System.out.println(in.next());
                    else
                              in.next(); // clear buffer that's not type int
      }
```

Practice 2

- Write a program that reads in 5 test scores from the user (handle invalid-type input)

  Example input: CS302 56.3 89.256 5.368

  Maidson HW2 5.36 12.3 3.58 1.258 6.358

  Store these in an array
- Output:
  - Minimum score
  - Maximum score
  - Average (mean) score

Arrays as Reference Variables

```
double[] testScores = {99, 95.6, 78, 82.9, 85.2, 88};
```

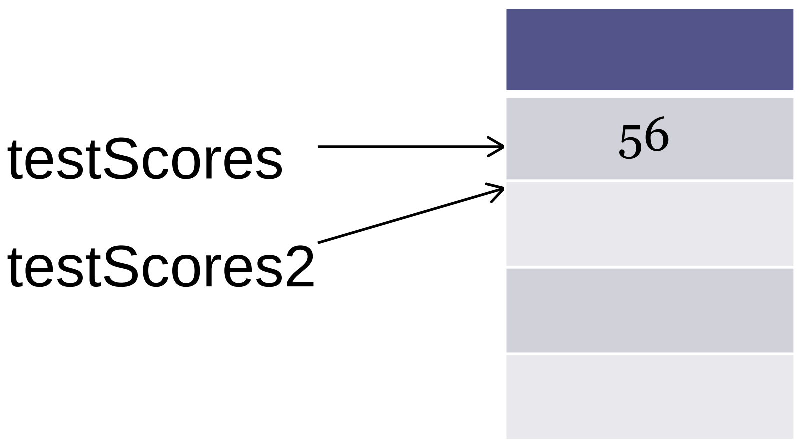
- testScores does not hold any of the actual test score values
- Instead it holds an address that points to where the array of values is in memory
- When you set one array equal to another, it does NOT create a new array, it simply sets the address pointer's to be equivalent
- Example: we have an int array named values1; if we do

  `int[] values2 = values1;` new copy or the same thing?

  Check out next slide!!

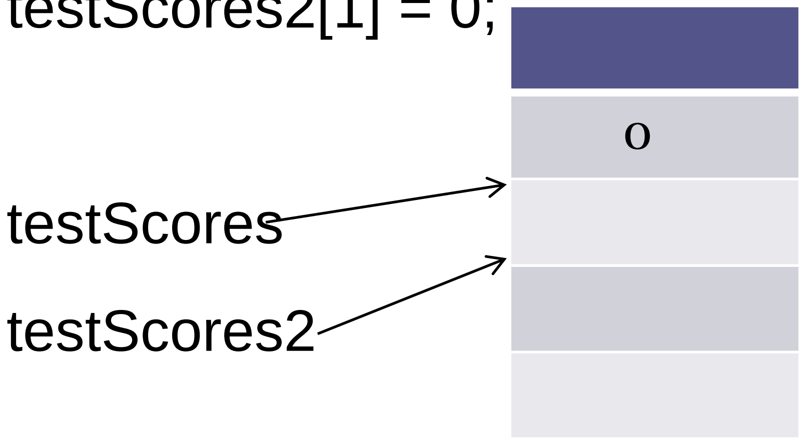
Arrays as References Example

```
double[] testScores = {99, 95.6, 78, 82.9, 85.2, 88};

double[] testScores2 = testScores;
```



```
//will print out 0

System.out.println(testScores[1]);
```

```
testScores2[1] = 0;
```

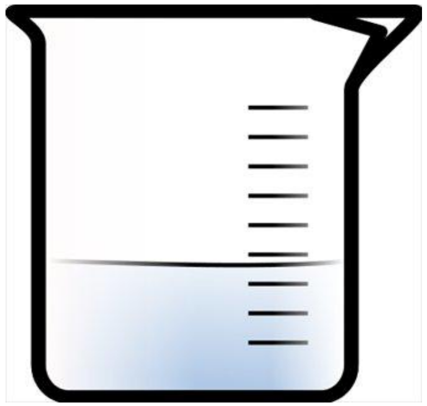


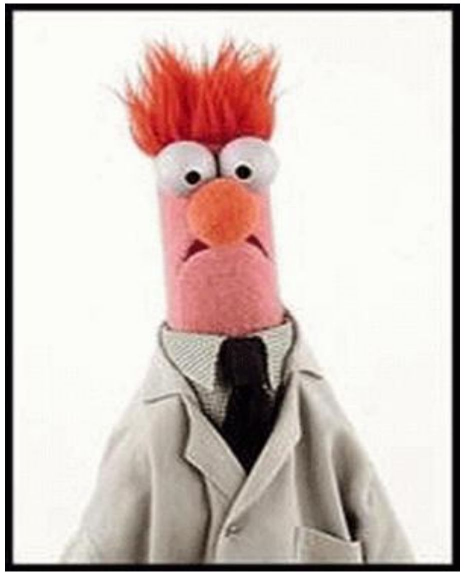
Array Limitations

- Fixed length
  - What if we get a new test score but we have filled up the array?
  - No way to simply add a new "box" for the score
  - Must create a new array that is long enough for all the old scores and the new score and copy all the values over

Partially Filled Arrays

- One solution to fixed-length problem: make array bigger than you would need and keep a counter of its current size (int currentSize)
- Why do we need a seperate variable? Why can't we just use the .length property?





Common Array Algorithms

- How would we code each of the following?
  - Printing out the elements of an array?
  - Find a specific value in an array?
  - Removing an element in an array?
    - If order doesn't matter?
    - If order does matter?
  - Insert an element at the end of an array?
  - Insert an element at a specified index in an array?

Copying Arrays

int[] testArray1 = {1, 2, 3};

- What value does testArray1 hold? (What type of variables are arrays?)
- What happens if I do:
  - int[] testArray 2 = testArray1;
- If we want to actually copy an array:
  - Arrays.copyOf(arrayToCopy, n);
    - n = how many elements to copy over – if n is < arrayToCopy.length will only copy first n; if n is > arrayToCopy.length will copy over the entire array and give you (n – arrayToCopy.length) extra indices

Copying Arrays Example (Maybe take home)

```
double[] data = {1, 2, 3};
double[] newData = Arrays.copyOf(data, data.length*2);
```

- How could we have done this without Arrays.copyOf?

Practice 3 (Maybe take home)

```
double[] data = {1, 2, 3};
double[] newData = Arrays.copyOf(data, data.length*2);
```

- How could we have done this without Arrays.copyOf?

Practice 4 (Maybe take home)

Given an array, how to get a new array that has elements with reverse order?

Practice 5: letter frequency (take home)

Read text (can be multiple lines) in from the standard input and count the frequency of each English letter that occurs. The end of the text is signalled by a line containing just the string "DONE.“

Example: <console input>: abcde fgh(now hit enter, user should be prompted again)

AbcDe FgH (now hit enter, user sh…….)

DONE (now program terminates since DONE is entered)

The result should be (‘a’ appears twice, ‘b’ appears twice, …

‘h’ appears twice)

THE UNIVERSITY of WISCONSIN MADISON