# Arrays of Primitives and Command Line Arguments

CSC 116 – Section 002

March 30, 2005



# Arrays

- "… a collection of data values of the same type." [Wu]
- Fixed size
- If you wish to change the size you need to create a new array
- All arrays are indexed starting at 0
- Can have an array of primitive data types: ints, doubles, booleans
- Can also have arrays of objects: Strings, Rectangles, BankAccounts

## Arrays (2)

- Arrays have a public constant that contains the length of the array, called *length*
- The length of an array is set when it is created
- We can use the length constant instead of a numeric value when we need to know the length.
- Call by: arrayName.length



## Creating Arrays

- Creating a new array where you will assign the elements individually:

  int [] data = new int[10];

  int data [] = new int[10];
- Creating a new array where you initialize elements at declaration:

  String [] monthNames = { “Jan”, “Feb”, “Mar”, “Apr”, “May”, “Jun”, “Jul”, “Aug”, “Sep”, “Oct”, “Nov”, “Dec” };

# Accessing Arrays

- Accessing an array element:

```
int fifthMember = data[4];
data[3] = 15;
data[i] = 5;
data[i – 2] = 4;
```



# Array Example

```
//Example of an array of ints
int data = new int[10]; //array declaration – size 10
for(int i = 0; i < data.length; i++) {
   data[i] = i; //assign value to array element i
}
```

# Copying Arrays

- Finish the following code snippet to copy array A to array B

```
int [] a = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };
int [] b = new int[10];
for(int x = ___; x < ___; x++) {
    b[ ___ ] = a[ ___ ];
}
```



# Copying Arrays Using arraycopy

- Use System.arraycopy(from,fromStart, to,toStart,count)
- from – source array
- fromStart – index position of source array to start copying from
- to – destination array
- toStart – index position of destination array to start copying too
- count – the number of elements to copy

## Copying Arrays Using arraycopy (2)

- Example:

```
int [] a = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };
int [] b = new int[10];
System.arraycopy(a,0,b,0,10);
```



## Copying Arrays Using arraycopy (3)

- Copy 5 elements from array A to array B starting at position 3 in array A and start copying to position 5 in array B.

```
int [] a = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };
int [] b = new int[10];
System.arraycopy(___,___,___,___,___);
```

# Arrays as Parameters

- Arrays can be used as parameters to constructors and methods just like other objects.



# Arrays as Parameters Example

```
public static double average(int [] data) {
   if(data.length = 0)
        return 0;
   double sum = 0;
   for(int x = 0; x < data.length; x++)
        sum += data[x];
   return sum / data.length;
}
```

# Arrays as Return Values

- Arrays can be used as return values just like other objects.



# Arrays as Return Values Example

```
public static int[] randomData(int length) {
   Random generator = new Random();
   int [] data = new int[length];
   for(int x = 0; x < data.length; x++) {
        data[x] = generator.nextInt(x);
   }
   return data;
}
```

# Command Line Options

- Command line options are strings typed in the command line when you tell the program to run
- Ex:

  java ProgramName opt1 opt2 opt3….



# Command Line Options (2)

- These options are passed into the program as an array of Strings
  - The command line option is split by a space
- Always check the length of the command line option array before accessing any of the options!

# Command Line Options Example

```
public class CommandLine {
   public static void main(String [] args) {
        for(int x = 0; x < args.length; x++) {
              System.out.println(args[x]);
        }
   }
}
```



# References

- Jason Schwarz's Lecture 19 & 20 slides: http://courses.ncsu.edu/csc116/