



Arrays

Problem Set: Assignment #1 due Thursday, February 9

C-1



Array Declarations

```
int[] arrayA;  
arrayA = new int[5];  
  
int[] arrayB = new int[5];  
  
int[] arrayC = {1, 2, 3, 4, 5};  
  
double[] arrayD = new double[10];  
  
char[] lettersArray = new char[5];  
  
String[] wordArray = {"CS230", "Data", "Struct"};
```

C-2



Accessing Array Elements

```
int[] arrayA;  
arrayA = new int[5];  
  
arrayA[0] = 5;  
arrayA[1] = 6;  
arrayA[2] = arrayA[0] + arrayA[1];
```

C-3



Accessing Array Elements

```
int[] arrayB = new int[5];  
for (int i = 0; i < 5; i++) {  
    arrayB[i] = 2*i;  
}  
  
int[] arrayC = {1, 2, 3, 4, 5};  
for (int i = 0; i < arrayC.length; i++) {  
    arrayC[i]++;  
    System.out.println(arrayC[i]);  
}
```

C-4



Accessing Array Elements

```
double[] arrayD = new double[10];
for (int i = 0; i < arrayD.length; i++) {
    arrayD[i] = 0.5*i;
}

char[] lettersArray = new char[5];
lettersArray[3] = 'x';

String[] wordArray = {"CS230", "Data", "Struct"};
wordArray[1] = "Silly ";
System.out.println(wordArray[1] + wordArray[2]);
```

C-5



Copying an Array

What happens here?

```
arrayA = arrayB;
```

What is printed here?

```
int[] arr1 = {1, 2, 3, 4, 5};
int[] arr2 = {1, 2, 3, 4, 5};
if (arr1 == arr2)
    System.out.println("same");
else
    System.out.println("different");
```

How do we copy the contents of arrayA into arrayB?
How do we check if two arrays contain the same info?

C-6



Methods can have Arrays as Input

```
public static int sumArray(int[] numArray) {
    int sum=0;
    for (int i=0; i<numArray.length; i++)
        sum = sum+numArray[i];
    return sum;
}

//... in main

int[] arrayA={1, 2, 3, 4, 5};
int result = sumArray(arrayA);
```

C-7



Methods can have Arrays as Output

```
public static int[] getNumArray(int size) {
    int[] newArray = new int[size];
    for (int i=0; i<size; i++)
        newArray[i] = i;
    return newArray;
}

//... in main

int[] arrayC = getNumArray(20);
```

C-8



Methods can Change Arrays as Side-Effects

```
public static void setArray(int[] numArray, int num) {
    for (int i=0; i<numArray.length; i++)
        numArray[i] = num;
    // does not explicitly return anything
}
```

//... in main

```
int[] arrayB = new int[10];
setArray(arrayB, 4);
```

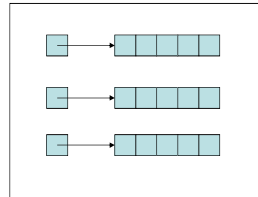
This means that arrays are "called by reference"

C-9



Let's Play "Computer"

Hand simulate the `main()` method in `ArrayExercise.java` and see what you get



C-10



What about those args?

The `String[] args` input parameter in the `main()` method is Java's way to communicate with the outside world at the time of invocation

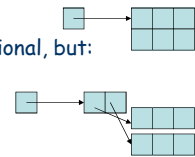
```
public class PlayGame {
    public static void main(String[] args) {
        String player1 = args[0];
        String player2 = args[1];
        System.out.print("Welcome to the game ");
        System.out.println(player1 + " and " + player2);
    }
}
```

C-11



Higher Dimension Arrays

You can think of them as multi-dimensional, but: in reality, they are arrays of arrays



```
int[][] table = new int[2][3];
table[1][2] = 100;
table[0][0] = -100;
```

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

```
int[] a, b, c; // a, b, and c all 1D arrays.
int a[], b, c[][]; // a is 1D; c is 2D; b is not an array.
```

C-12



Arrays vs. Objects

Like objects

- Value stored in array variable is a pointer - not the real thing
- Use `new` to create it
- Has instance variable length
- It passes pointer to/from methods

Different than objects

- No class name for arrays
- Collection of same type, referenced by index
- Can't create instance methods invoked with an array

C - 13



The Vector Class

- The `Vector` class implements a growable array of objects
- Like an array, it contains components that can be accessed using an integer index
- The size of a `Vector` can grow or shrink as needed to accommodate adding and removing items after the `Vector` has been created

C - 14