

Arrays and Methods

When objects, such as arrays, are used as a parameter in a method what actually gets passed is a reference to the object and not the object itself.

The actual and formal parameters refer to the same object. Any changes the method makes to the object changes the original object and not a copy of the object.

Example #1

Method called from main method - Sum the elements of an array

```
private static int sum (int[] a) {  
    int result = 0;  
    for (int x=0; x<a.length; x++) {  
        result = result + a[x];  
    }  
    return result;  
}
```

Call to the method from the main method

```
int[] array1 = {10, 24, 16, 78, -55, 89, 65};  
int[] array2 = {4334, 22928, 33291};  
  
if (sum(array2) > sum(array1)) {  
    System.out.println("Hello World");  
}
```

Example #2

Method called from main method - Search for a Value

```
private static int search (int[] a, int searchValue) {  
    for (int x=0; x<a.length; x++){  
        if (a[x]==searchValue) return x;  
    }  
    return -1;  
}
```

Call to the method from the main method

```
int[] code = {2267, 4281, 1498, 8923, 6562};  
String[] name = {"Bob", "Mary", "Frank", "Marla", "John"};  
  
int temp = search(code, 1498);  
  
if (temp!=-1 ) {  
    System.out.println(name[temp]);  
}
```

Example #3

Method called from main method - Sum the Rows

```
private static int[] sumRows(int[][] a) {
    int[] rowSum = new int[a.length];
    for (int x = 0; x < a.length; x++) {
        for (int y = 0; y < a[x].length; y++) {
            rowSum[x] = rowSum[x] + a[x][y];
        }
    }
    return rowSum;
}
```

Call to the method from the main method

```
int[][] twoD = {{1,2,3,4}, {5,6}, {7,8,9}};
int[] oneD;

oneD = sumRows(twoD);
```

Example #4

Method called from main method - Tries to Copy An Array but Fails

```
private static void copyOne(int[] original, int[] copy) {  
    copy = new int[original.length];  
    for (int x = 0; x < original.length; x++) {  
        copy[x] = original[x];  
    }  
}  
}
```

Call to the method from the main method - Does not give desired results

```
int[] orig = {1,2,3,4,5};  
int[] cp;  
copyOne(orig, cp);
```

When copyOne terminates, cp should be a copy of copy(original). But the original variable cp is not changed by the call to the method copyTwo().

Example #5

Method called from main method - Copy An Array

```
private static int[] copyTwo(int[] original) {  
    int[] copy = new int[original.length];  
    for (int x = 0; x < original.length; x++) {  
        copy[x] = original[x];  
    }  
}  
return copy;  
}
```

Call to the method from the main method

```
int[] orig = {1,2,3,4,5};  
int[] cp = copyTwo(orig);
```

Example #6

Main Menu - Array of Objects

```
Student[] studentArray = new Student[3];

studentArray[0] = new Student("Jeff", 90, 100, 50);
studentArray[1] = new Student("Mary", 100, 95, 98);
studentArray[2] = new Student("Susie", 100, 100, 100);

int sum = 0;
for (int x = 0; x < studentArray.length; x++) {
    sum = sum + studentArray[x].getAverage();
}
System.out.println("Class average: " + sum/studentArray.length);
```

Note: The Student object is the object based on the Student class from chapter 5. The new Student object from chapter 9 contains an array of test scores not just three.

