Methods and References

CS 1324

Tip of the Day

- One way to practice with this material is to write simple programs that use parameter passing and returning of arrays
- Trace the program with memory diagrams to see what it should do
- Run it to verify your answer
- When you get to the point that you can accurately predict what will happen, you've mastered the material

Recall: Passing Primitive Data

```
// Calling method
int x = 3;
int y = 7;
swap(x,y);

public static void swap(int a, int b)
{
    int temp = a;
    a = b;
    b = temp;
}
```

Observation

- No primitive data type can ever be changed inside a method because….
- Remember: Called pass by value

Array Parameter Passing

- Three cases:
  - What happens if you change the contents of the array
  - What happens if you reconstruct an array inside a method
    - Return the reference
    - This reference may be assigned or not
  - What happens if you reconstruct an array inside a method
    - Do not return the reference
- Use memory diagrams

Change Contents of Array

```
// main program
int[] data = {1, 2, 3, 4, 5};
rotateLeft(data);

public static void rotateLeft(int[] source)
{
    if (source.length<=1) return;
    int temp = source[0];
    for (int i=0; i<source.length-1; ++i)
            source[i] = source[i+1];
    source[source.length-1] = temp;
}
```

Articulate a Rule

- What happens when array contents are changed, but the array is not reallocated?
- This is called passing by sharing
- Why did this happen?

Instant quiz Question 1

```
int[] data = {1, 1, 1};
mystery(data, 3);

public static void mystery (int[] source, int value) {
source[1] = value;
}
```

What will data contain after the method call?

a) {1, 1, 1} b) {1, 3, 1} c) {3, 3, 3} d) {3, 1, 1}

Reconstruct, but not returned

```
// main program
int[] data = {1, 2, 3, 4, 5};
rotateLeft(data);
public static void rotateLeft(int[] source) {
    if (source.length==0) return;
    int[] temp = new int[source.length];
    for (int i=0; i<source.length-1; ++i)
        temp[i] = source[i+1];
    temp[source.length-1] = source[0];
    source = temp; }
```

Articulate a Rule

- What happens when the array is reallocated in the method, but not returned

- This is a bug, so it doesn’t have a cool name

Instant Quiz Question 2

```
int[] data = {1, 1, 1};
mystery(data, 3);
public static void mystery (int[] source, int value) {
    source = new int[3];
    source[0] = value;
    source[1] = value;
    source[2]= value;
}
```

What will data contain after the method call?

a) {1, 1, 1}   b) {1, 3, 1} c) {3, 3, 3} d) {3, 1, 1}

Reconstruct and Return

```
// main program
int[] data = {1, 2, 3, 4, 5};
data = rotateLeft(data);
public static int[] rotateLeft(int[] source) {
    if (source.length<= 1) return source;
    int[] temp = new int[source.length];
    for (int i=0; i<source.length-1; ++i)
        temp[i] = source[i+1];
    temp[source.length-1] = source[0];
    return temp;}
```

Articulate a Rule

- What happens when the array is reallocated in the method and returned?
- What if the return value is assigned to the same array reference?
  - data = rotateLeft(data);
- What if the return value is assigned to a different array reference?
  - int[] target = rotateLeft(data);
- What if the return value is not assigned to any array reference?
  - rotateLeft(data);

Instant Quiz Question 3

```
int[] data = {1, 3, 5, 7, 9};
mystery(data);

public static int[] mystery(int[] data){
    data[0] = 3;
    data = new int[3];  // gets initialized to zero
    return data;
}
```

After the method call data contains:

a) {3, 3, 5, 7, 9}   b) {1, 3, 5, 7, 9}

c) {0, 0, 0}   d) Something else

Observation

- The reason that parameter passing for arrays works this way is that array references are passed to methods
- Object references are also passed to methods
- Object parameter passing has exactly the same rules as arrays
  - Complicated by the existence of objects like String that cannot be changed (immutable)

Passing Immutable Objects

```
// Calling Method
String word = new String(“abcde”);
repeat(word);

public static void repeat(String source)
{
    source = source + source;
}
```

Returning Immutable Objects

```
// Calling Method
String word = new String(“abcde”);
word = repeat(word);

public static String repeat(String source)
{
    source =  source + source;
    return source;
}
```

Instant Quiz Question 4

- What is in the String name after the method call?

```
// Called in main method
String name = new String("Raven");
erase(name);
// Method
public static String erase(String data) {
    data = "";
    return data;}
```

a) null b) An empty Stringc) "Raven"

d) none of the above