CS 220 Software Design II — Spring 2022

In-class Exercises 08

University of Wisconsin - La Crosse — *Date: April 15*

1. For the algorithm below, 1) annotate each line of code **just above** the line to show the cost of that particular line, 2) write out the full expression demonstrating the cost of the algorithm in the space below the code, and 3) simplify to the big O expression in the space below the code.

```
1  // assume int[][] arr2d was declared/instantiated/filled with numbers
2  // do not include the set up of arr2d in your calculation
3  // n
4  for(int i = 0; i < arr2d.length; i++) {
5
6      // 1
7      if(arr[i].length > 10) {
8
9          // 1
10         System.out.println("Lots of numbers here.");
11
12     // 0
13     } else {
14
15         // c (c is a constant ≤ 10)
16         for(int j = 0; j < arr2d[i].length; j++) {
17
18             // c
19             arr[i][j] *= 10;
20
21             // c
22             System.out.print(arr[i][j] + " ");
23         }
24     }
25
26     // 1
27     System.out.println();
28 }
```

Solution:

n * ((c * (1 + 1 + 1)) + 1)
n * (3c + 1)
O(n)

2. What causes a stack overflow error? Your description should address **what would be wrong in the code** and **what happens to the stack**.

Solution:

1) infinite recursion due to incorrect/lack of base case
2) stack runs out of room

3. Describe one downside to recursive programming when compared to iterative programming in terms of memory.

Solution:

takes up increasingly more room on the stack

4. Write a `public static` non-void method called `noVowels` described in the Javadoc comment below. Your solution should be recursive and only use conditionals and basic `String` operations; loops are **not** allowed.

```
/**
 * Takes a lower-case String as a parameter and returns a String in which
 * the vowels have been removed:
 *      noVowels("")  => ""                          noVowels("xyz") => "xyz"
 *      noVowels("abcdef")  => "bcdf"      noVowels("aeiou") => ""
 * @param s  A Java String
 * @return A Java String containing only non-vowel characters from s in the
 *         order they appear in s
 */
```

Solution:

```
public static int sumDigits(int n) {
    if (s.length() == 0) {
        return s;
    } else {
        char first = s.charAt(0);
        s =  s.substring(1);
        if(first == 'a' || first == 'e' || first == 'i' || first == 'o'
                        || first == 'u')
            return noVowels(s);
        else
            return first + noVowels(s);
    }
}
```

5. Write a `public static void` method called `recurFizzBuzz` that takes in an `int` n and then prints each number to the console on its own line, all the way to (and including) 1. If a number is divisible by 3, instead of printing the number, the method should print "fizz"; if the number is divisible by 5, instead of printing the number, the method should print "buzz"; and if the number is divisible by 3 and 5, instead of printing the number, the method should print "fizz buzz". **Your solution must be recursive to receive credit.**

```
/**
 * Example (n = 15):
 * fizz buzz
 * 14
 * ... (13-6 omitted due to space)
 * buzz
 * 4
 * fizz
 * 2
 * 1
 * @param n A non-negative integer
 */
```

Solution:

```
public static void recurFizzBuzz(int n) {
    if(n % 3 == 0) {
        System.out.print("fizz");
    }
    if(n % 5 == 0) {
        System.out.print("buzz");
    }
    if(n % 3 != 0 && n % 5 != 0) {
        System.out.print(n);
    }
    System.out.println();
    if(n == 1) {
        return;
    } else {
        recurFizzBuzz(n-1);
    }
}
```