**Recursion – Tracing**

1. Given the method defined below, what does the following return: mystery(5)?

public static int mystery(int n) {

if (n == 0)

return 1;

else

return 2 \* mystery (n - 1);

}

Mystery 5 => 2\*mystery(5-1) => 2\*2\*mystery(3-1)

At this point, it is easy to deduce that mystery n is 2^n,

Therefore mystery 5 = 2^5 = **32**

2. Given the method defined below, what does the following print: mystery(4, 3)?

public static int mystery(int n, int a) {

if (n == 1)

return a;

return a \* mystery(n-1,a);

}

Mystery(n,a) = a\*(n-1,a), and a^n = a(a^n-1), therefore mystery(n,a) = a^n, mystery(4,3) = 3^4 = **81**

3. Given the method defined below, what does the following return: bunnyEars(5)?

public static int bunnyEars(int bunnies) {

if (bunnies == 0)

return 0;

else if (bunnies == 1)

return 2;

else

return 2 + bunnyEars(bunnies - 1);

}

bunnyEars(n) = 2+(n-1), and n\*2=2n, therefore

bunnyEars(n) = 2n, bunnyEars(5) = 2\*5 = **10**

4. Given the method defined below, what does the following print: mystery(1234)?

public static void mystery (int x) {

System.out.print(x % 10);

if ((x / 10) != 0) {

mystery(x / 10);

}

System.out.print(x % 10);

}

| Pass 1: mystery(1234) | 4 + mystery(123) + 4 |
| --- | --- |
| Pass 2: mystery(123) | 4+3+mystery(12)+3+4 |
| Pass 3: mystery(12) | 4+3+2+mystery(1)+2+3+4 |
| Pass 4: mystery(1) | 4+3+2+1+1+2+3+4 |

Therefore Mystery(1234) = 43211234

5. Given the method defined below, what does the following return: mystery(“xyzxyxy”)?

public static int mystery(String str){ if (str.length() == 1)

return 0;

else {

if (str.substring(0,1).equals("y"))

return 1 +mystery(str.substring(1));

else

return mystery(str.substring(1)); }

}

Pass 1: mystery(“xyzxyxy”)

Pass 2: mystery(“yzxyxy”)

Pass 3: 1 + mystery(“zxyxy”)

Pass 4: 1 + mystery(“xyxy”)

Pass 5: 1 + mystery(“yxy”)

Pass 6: 1 + 1 + mystery(“xy”)

Pass 7: 1 + 1 + mystery(“y”)

Pass 8: 1 + 1 + 0 = 2