Quick Write – Recursion Tracing
CS159 Spring 2024 Name: _____

Trace each of the following recursive routines using the method demonstrated in class.

1. powerN

```
public static void main(String[] args) {
    System.out.println("powerN(2, 3): " + powerN(2, 3));
}
public static int powerN(int base, int n) {
    if (n == 0) {
        return 1;
    }
    return base * powerN(base, n - 1);
}
```

- What is the return result?
- What is the maximum depth of the call stack of recursive calls?
- How many times was the recursive method called?

2. count7

```
public static void main(String[] args) {
    System.out.println("count7(1772): " + count7(1772));
}
public static int count7(int n) {
    if (n < 10) {
        if (n == 7) {
            return 1;
        } else {
            return 0;
        }
    }
    if (n % 10 == 7) {
        return 1 + count7(n / 10);
    } else {
        return count7(n / 10);
    }
}
```

- What is the return result?
- What is the maximum depth of the call stack of recursive calls?
- How many times was the recursive method called?

3. choose

```
public static void main(String[] args) {
    System.out.println("choose(4, 2): " + choose(4, 2));
}
public static int choose(int n, int k) {
    if (k == 0 || k == n) {
        return 1;
    }
    return choose(n - 1, k - 1) + choose(n - 1, k);
}
```

- What is the return result?
- What is the maximum depth of the call stack of recursive calls?
- How many times was the recursive method called?

4. challenge1

```
public static void main(String[] args) {
    System.out.println("challenge1(345): " + challenge1(345));
}

public static int challenge1(int n) {
    if (n == 0) {
        return 0;
    }
    int last = n % 10;
    return last + challenge1(n / 10);
}
```

- What is the return result?
- What is the maximum depth of the call stack of recursive calls?
- How many times was the recursive method called?
- What does this code do? Give a general description.

5. challenge2

```
public static void main(String[] args) {
  System.out.println(
          "challenge2(Hello, l, e): " + challenge2("Hello", 'l', 'e'));
}

public static String challenge2(String str, char c1, char c2) {
    if (str.length() == 0) {
        return str;
    }
    if (str.charAt(0) == c1) {
        return c2 + challenge2(str.substring(1), c1, c2);
    }
    return str.charAt(0) + challenge2(str.substring(1), c1, c2);
}
```

- What is the return result?
- What is the maximum depth of the call stack of recursive calls?
- How many times was the recursive method called?
- What does this code do? Give a general description.

6. challenge3

```
public static void main(String[] args) {
    int[] nums2 = {12, 2, 12, 1, 3, 12};
    System.out.println(
            "challenge3(nums2, 0, 12): " + challenge3(nums2, 0, 12));
}
public static int challenge3(int[] nums, int i1, int i2) {
    if (i1 == nums.length) {
        return 0;
    }
    if (nums[i1] == i2) {
        return 1 + challenge3(nums, i1 + 1, i2);
    }
    return 0 + challenge3(nums, i1 + 1, i2);
}
```

- What is the return result?
- What is the maximum depth of the call stack of recursive calls?
- How many times was the recursive method called?
- What does this code do? Give a general description.