CSCI-2725 Homework 2

Yitao Tian

March 3, 2023

Extra credit: There is 5 percentage extra credit if you don't submit hand-written homework. You can use LaTeXor any other tool to write your homework.

1. (5 points each) Look at the pseudocode below, and find the Big-*O* estimate or worst case complexity of the following recursive functions.

   Hint: Write recurrence relation for the functions and solve it using Master's Theorem.

```
1  void fun(int n, int m) {
2    if (m > n) return;
3    System.out.print("m = " + m);
4    fun(n, m + 2);
5  }
```

```
float fun(int a[], int i) {
  if (i == 0) {
    return a[0];
  }
  if (i > 0) {
    return (i * fun(a, i - 1) + a[i])/(i + 1);
  }
}
```

Assume that the function "random" has a complexity of $O(n)$.

```
1  void fun(int array[], int a, int b) {
2    if (a > b) return;
3    mid = (a + b)/2;
4    fun(array, a, mid);
5    fun(array, mid + 1, b);
6    random(array, a, mid, b);
7  }
```

```
int fun(int x, int n) {
  if (n == 0) return 1;
  if (x == 0) return 0;
  if (n == 1) return x;
  if (n % 2 == 0) return fun(x * x, n / 2);
  else return x * fun(x * x, n / 2);
}
```

2. (10 points) Consider a recurrence relation given by following.

$$T(n) = 2T(n-1) + 2^n$$

Show that $T(n) = n2^n$ is the solution of the above equation. If you need a base case, use $T(0) = 0$.

3. (10 points) Below is the code for Fibonacci sequence. Write a recurrence relation for this code and solve it using substitution (forward or backward) method.

   Note: There are two recurrence relations $T(n-1)$ and $T(n-2)$ and to simplify your calculation you can assume that $T(n-1) = T(n-2)$ as they are almost of same size.

```
1   int Fibonacci(int N) {
2     if (n == 0 || n == 1) {
3       return 1;
4     }
5     else {
6       return Fibonacci(n - 1) + Fibonacci(n - 2);
7     }
8   }
```