Name: Michael Rogers
ID: 105667404

CSCI 3104, Algorithms **Profs. Grochow & Layer**
Problem Set 2 **Spring 2019, CU-Boulder**

Hyperlinks for convenience: 1a 1b 1c 1d 2 3a 3b 3c 3d

1. (20 pts total) *Solve the following recurrence relations using any of the following methods: unrolling, tail recursion, recurrence tree (include tree diagram), or expansion. Each case, show your work.*

   (a) $T(n) = T(n-4) + C\,n$ if $n > 1$, and $T(n) = C$ otherwise

   Unrolling Gives us:

   $T(n-4) = T(n-8) + C(n-4)$
   $T(n-8) = T(n-12) + C(n-8)$
   $T(n-12) = T(n-16) + C(n-12)$
   .
   .
   .
   $T(4) = T(0) + 4C$
   $T(n) = T(0) + C(4 + 8 + 12 + ... + n)$
   $T(n) = C + C(1 + 2 + 3 + ... + \frac{n}{4})$
   $T(n) = C + 4C(\frac{n}{4})(\frac{\frac{n}{4}+1}{2}) \Rightarrow C(1 + n(\frac{n+4}{8})$

   $\Rightarrow T(n) = \Theta(n^2)$

(b) $T(n) = 3T(n-2) + 1$ if $n > 1$, and $T(n) = 3$ otherwise

(c) $T(n) = T(n-1) + 2^n$ if $n > 1$, and $T(1) = 3$

(d) $T(n) = T(n^{1/2}) + 1$ if $n > 2$, and $T(n) = 0$ otherwise

2. (10 pts) *Consider the following function:*

```
def foo(n) {
     if (n > 1) {
        print( ''hello'' )
        foo(n/3)
        foo(n/3)
        foo(n/3)
     }
}
```

In terms of the input n, determine how many times is "hello" printed. Write down a recurrence and solve using the Master method.

I don't know

Name: Michael Rogers
ID: 105667404

3. (30 pts) *Professor Flitwick asks you to help him with some arrays that are slumped. An array A is slumped if $A[1..i]$ has the property that, for some $C > 0$, $A[j+1] = A[j] - C$ for $1 \leq j < i$, and $A[i..n]$ has the property that, for some $D > 0$ where $C \neq D$, $A[j+1] = A[j] + D$ for $i \leq j < n$. Using his wand, Flitwick writes the following slumped array on the board $A = [7, 3, -1, -5, 0, 10, 15, 20, 25]$, as an example.*

   (a) *Flitwick found that one of his slumped arrays had an identical adjacent value (i.e., $A[j] = A[j+1]$) and no longer trusts any of his slumped arrays. Write a recursive algorithm that takes asymptotically sub-linear time to ensure that there are no identical adjacent elements in A.*

   I don't know

(b) *Prove that your algorithm is correct. (Hint: prove that your algorithm's correctness follows from the correctness of another correct algorithm we already know.)*

I don't know

Name: Michael Rogers
ID: 105667404

(c) *Now consider the multi-slumped generalization, in which the array contains k local minima, i.e., it contains k subarrays, each of which is itself a slumped array. Let $k = 2$ and prove that your algorithm can fail on such an input.*

I don't know

(d) *Suppose that $k = 2$ and we can guarantee that neither local minimum is closer than $n/3$ positions to the middle of the array, and that the "joining point" of the two singly-slumped subarrays lays in the middle third of the array. Now write an algorithm that tests A for identical adjacent values in sublinear time. Prove that your algorithm is correct, give a recurrence relation for its running time, and solve for its asymptotic behavior.*

I don't know