
DSC 40B - Discussion 04

Problem 1.

- a) State (but do not solve) the recurrence relation describing this function's run time.

```
import random
def foo(n):
    if n <= 2:
        return
    for i in range(n):
        for j in range(i,n):
            print(i)
    return foo(n//2) + foo(n//2)
```

- b) Suppose a binary search is performed on the following array using the implementation of *binary_search* from lecture. What is the worst case number of equality comparisons that would be made to search for an element in the array? That is, what is the greatest number of times that `arr[middle] == target` can be run?

[1, 4, 7, 8, 8, 10, 15, 51, 60, 65, 71, 72, 101]

- c) State (but do not solve) the recurrence relation describing this function's run time.

```
def fibonacci(n):
    if n == 0:
        return 0
    if n == 1:
        return 1
    return fibonacci(n - 1) + fibonacci(n - 2)
```

How can we improve the run time of this function?

Below is some sample code for the improved function.

```
def fib_improved(n):
    if n == 0:
        return (0, 0)
    if n == 1:
        return (1, 0)
    (a, b) = fib_improved(n - 1)
    return (a + b, a)
```

The new recurrence relation is $T(n) = c + T(n - 1)$, which is $\Theta(n)$.

Problem 2.

Solve the following recurrence relations.

- a) $T(n) = T(n-1) + n$
 $T(0)=0$

b) $T(n) = 4T(n/4) + n$
 $T(1) = 1$