Matthew Kramer

Homework 6

Due 09/13/16

September 8, 2016

1. Analyze the *worst-case* time complexity of the algorithm below. Please show all work. The $\lfloor\ \rfloor$ symbols represent the *floor* ("round down") function. You may assume that this function takes $\Theta(1)$ time for any input. You may also assume it takes a constant amount of time to determine whether an integer is odd.

   Note that figuring out what problem this algorithm solves is *irrelevant* to analyzing its complexity.

```
        Input: n: nonnegative integer
        Algorithm: LoopMystery
Θ(1)    sum = 0
Θ(1)    t = 1
Θ(1)    d = 1
Θ(1)    k = n
        while k > 1 do
            for i = 1 to k do              }
Θ(1)            t = t + d                  }  n-1-1 iterations = Θ(n-2) = Θ(n)
Θ(1)            sum = sum + t              }
            end
Θ(1)        if k is odd then
Θ(1)            d = -d
            end
O(1)        k = ⌊k/2⌋
        end
Θ(1)    return sum
```

$$\sum_{i=1}^{\lg(n)} \frac{n}{2^i} = \frac{n}{2^1} + \frac{n}{2^2} + \frac{n}{2^3} + \ldots + \frac{n}{2^{\lg(n)}}$$

$$= \frac{n}{2^1} + \frac{n}{2^2} + \frac{n}{2^3} + \ldots + \frac{n}{n}$$

$$= \underbrace{\frac{n}{2^1} + \frac{n}{2^2} + \frac{n}{2^3} + \ldots + 1}_{n}$$

$\rightarrow O(n) + O(1)$

$\rightarrow O(n)$