Homework **3**

Due 09/23/19

September 16, 2019

1. Find the *worst-case* complexity for the algorithm below. You may assume that n is a power of 2. *Hint:* The worst-case behavior occurs when $t \notin data$.

```
   Input: data: sorted array of n integers
   Input: n: size of data
   Input: t: an integer
   Output: index x such that data[x] = t, or 0 if t ∉ data
 1 Algorithm: BinSearch
 2 lo = 0
 3 hi = n
 4 while hi − lo > 1 do
 5 |   mid = ⌈(hi + lo)/2⌉
 6 |   if data[mid] = t then
 7 |   |   return mid
 8 |   else if data[mid] > t then
 9 |   |   hi = mid
10 |   else
11 |   |   lo = mid
12 |   end
13 end
14 if data[hi] = t then
15 |   return hi
16 else
17 |   return 0
18 end
```

O(1) constant

O(log n) divides in half, log n times in the Worst Case

O(1)

BinSearch = O(log n)

2. Give a recurrence that describes the worst-case complexity of the RecursiveContains algorithm below. Justify your answer. You may assume that adding an element to an array takes constant time.

```
   Input: data: array of n integers
   Input: n: size of data
   Input: t: an integer
   Output: whether t ∈ data
 1 Algorithm: RecursiveContains
 2 if n = 0 then
 3 |   return false
 4 end
 5 large = {}
 6 small = {}
 7 for i = 2 to n do
 8 |   if data[i] < data[1] then
 9 |   |   Add data[i] to small
10 |   else if data[i] > data[1] then
11 |   |   Add data[i] to large
12 |   end
13 end
14 if data[1] = t then
15 |   return true
16 else if data[1] < t then
17 |   return RecursiveContains(large, t)
18 else
19 |   return RecursiveContains(small, t)
20 end
```

O(1) Constant

O(n)

O(1)

In the Worst Case, all data goes into either large or small and then we recur with n-1

T(n-1) + O(n)

3. Suppose that $T(n) = 8T(n/2) + f(n)$.

(a) What must be true of $f(n)$ in order for $T(n)$ to be $\Theta(n^3)$? Justify your answer.

(c) What must be true of $f(n)$ in order for $T(n)$ to be $\Theta(n^2)$? Justify your answer.

V1 case 1

$a = 8, b = 2, c = \log_2 8 = 3$

A.) If $n^c = n^3$ then

$f(n) = \Theta(n^{c-\epsilon})$ for some $\epsilon > 0$.

V2 case 2

B) n^3 vs $f(n)$?

for $T(n)$ to be $\Theta(n^2)$,

$f(n) = n^c$, with $c > \log_b a$

Since n^2 means

$c = 2 \not> 3 = \log_b a$, so

this is impossible