CSCI 3104 Spring 2018 Problem Set #3
Alison Ostlund (05/22)

---

1. (5 pts total) For parts (1a) and (1b), justify your answers in terms of deterministic QuickSort, and for part (1c), refer to Randomized QuickSort. In both cases, refer to the versions of the algorithms given in Lecture 3.

   (a) What is the asymptotic running time of QuickSort when every element of the input A is identical, i.e., for $1 \leq$ i, j $\leq$ n, A[i] = A[j]?

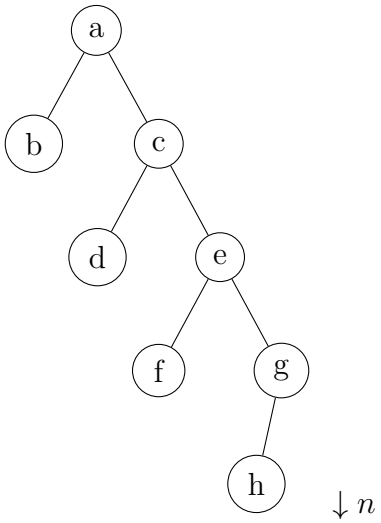
   Worst case for quick sort would be the case the array is all the same number for each position ex: [2,2,2,2,2,2,2,2]. The quick sort algorithm would only be calling partition on one element at a time. $T(n) = T(n1) + \Theta(n)$
   $T(n) = T(n2) + \Theta(n1) + \Theta(n)$
   $T(n) = T(n3) + \Theta(n2) + \Theta(n1) + \Theta(n)$
   $\vdots$
   $T(n) = \Theta(1) + \ + \Theta(n2) + \Theta(n1) + \Theta(n)$
   $T(n) = \sum_{i=1}^{n} \Theta(n) = \frac{n(n-1)}{2} = \Theta(n^2)$

   (b) Let the input array A = [9, 7, 5, 11, 12, 2, 14, 3, 10, 6]. What is the number of times a comparison is made to the element with value 3?

```
[9,7,5,11,12,2,14,3,10,6] - A[i]=9 A[j]=7 A[r]=6
[9,7,5,11,12,2,14,3,10,6] - A[i]=9 A[j]=5
[9,5,7,11,12,2,14,3,10,6] - A[i]=5 A[j]=7
[9,5,7,11,12,2,14,3,10,6] - A[i]=5 A[j]=11
[5,9,7,11,12,2,14,3,10,6] - A[i]=9 A[j]=12
[5,9,7,11,12,2,14,3,10,6] - A[i]=9 A[j]=2
[9,5,2,11,12,7,14,3,10,6] - A[i]=2 A[j]=14
[5,2,9,11,12,7,14,3,10,6] - A[i]=2 A[j]=3
[|5,2,3|,11,12,7,14,9,10,6] - A[i]=2 A[j]=3 A[r]= 3
[5,2,3]
Takes three times.
```

(c) How many calls are made to random-int in (i) the worst case and (ii) the best case? Give your answers in asymptotic notation.

Worst Case for RQS: $\Theta(n)$ This is because given an array that is already sorted, the BST will have the height of n with one leaf at each node. This will mean the worst case will be n times.
BST with height n:



Best case: $\Theta(\lg(n))$
The best case of RQS will have a tree with height lg(n). Therefore random-int will be called $\Theta(\lg(n))$

2. (30 pts total) Professor Trelawney has acquired n enchanted crystal balls, of dubious origin and dubious reliability. Trelawney needs your help to identify which crystal balls are accurate and which are inaccurate. She has constructed a strange contraption that fits over two crystal balls at a time to perform a test. When the contraption is activated, each crystal ball glows one of two colors depending on whether the other crystal ball is accurate or not. An accurate crystal ball always glows correctly according to whether the other crystal ball is accurate or not, but the glow of an inaccurate crystal ball cannot be trusted. You quickly notice that there are four possible test outcomes:

crystal ball i glows	crystal ball j glows		
red	red	$\Rightarrow$	at least one is inaccurate
red	green	$\Rightarrow$	at least one is inaccurate
green	red	$\Rightarrow$	at least one is inaccurate
green	green	$\Rightarrow$	both accurate, or both inaccurate

(a) Prove that if n/2 or more crystal balls are inaccurate, Trelawney cannot necessarily determine which crystal balls are accurate using any strategy based on this kind of pairwise test. Assume a worst-case scenario in which the inaccurate crystal balls contain malicious spirits that collectively conspire to fool Trelawney.

In the case that there is a set of half accurate and another half of inaccurate crystal balls, can cut the set until you get down to one accurate and one inaccurate ball, which causes there to be no other elements to compare the two balls to in order to determine their accuracy with the information given.

(b) Suppose Trelawney knows that more than n/2 of the crystal balls are accurate, but not which ones. Prove that bn/2c pairwise tests are sufficient to reduce the problem to one of nearly half the size.

```
Using the table given:
G = Green   R = Red
if two balls are accurate: GG
if two balls are inaccurate: GG
if one ball accurate and one inaccurate: RR

When going through a set of balls and comparing two
balls at a time,[i, j+1] when i, j = 1 use this set
of rules to determine accuracy of balls. If two
similar colors, remove one element, keep traversing.
```

```
When two different colors, remove both elements then
keep traversing. Using this method you are to either
get rid of 1 of 2 balls or both balls. This will
cause n to be cut in half(n/2) or sometimes cut by more.
This method will maintain the property that accurate
balls > inaccurate balls because as stated above if
there are two reds that means you get rid of one ball
but if you have two different balls you get rid of
both.
```

(c) Now, under the same assumptions as part (2b), prove that all of the accurate crystal balls can be identified with $\Theta(n)$ pairwise tests. Give and solve the recurrence that describes the number of tests.
$T(n) = T(\frac{n}{2}) + \frac{n}{2}$
$= T(\frac{n}{4}) + \frac{n}{2} + \frac{n}{4}$
$= T(\frac{n}{8}) + \frac{n}{2} + \frac{n}{4} + \frac{n}{8}$
$= \sum_{i=1}^{lg(n)} \frac{n}{2} = n - 1 \Rightarrow \Theta(n)$

3. (20 pts) Professor Dumbledore needs your help. He gives you an array A consisting of n integers A[1], A[2], . . . , A[n] and asks you to output a two-dimensional n n array B in which B[i, j] (for i ¡ j) contains the sum of array elements A[i] through A[j], i.e., B[i, j] = A[i] + A[i + 1] + + A[j]. (The value of array element B[i, j] is left unspecified whenever i ≥ j, so it doesnt matter what the output is for these values.) Dumbledore suggests the following simple algorithm to solve this problem:

```
dumbledoreSolve(A) {
for i = 1 to n {
   for j = i+1 to n {
       s = sum(A[i..j]) // look very closely here
       B[i,j] = s
}}}
```

(a) For some function g that you should choose, give a bound of the form $\Omega(g(n))$ on the running time of this algorithm on an input of size n (i.e., a bound on the number of operations performed by the algorithm).

$$\sum_{i=1}^{n}(\sum_{j=i+1}^{n}(j-i+1)) = \frac{1}{2}\sum_{i=1}^{n}(i-n-3)(i-n)$$
$$= \frac{1}{6}n(n^2+3n-4) = \frac{n^3+3n^2-4n}{6}$$
$$\frac{1}{6}n^3 + \frac{n^2}{2} - \frac{2n}{3} \Rightarrow \Theta(n^3) \therefore \Omega(n^3)$$

(b) For this same function g, show that the running time of the algorithm on an input of size n is also O(g(n)). (This shows an asymptotically tight bound of $\Theta(g(n))$ on the running time.)

$$\sum_{i=1}^{n}(\sum_{j=1}^{2} j) > \sum_{i=1}^{n}(\sum_{j=i+1}^{n}(j-i+1))$$
$$\sum_{i=1}^{n}(\sum_{j=1}^{2} j) = \frac{1}{2}\sum_{i=1}^{n} n^3 + n + 1$$
$$\Rightarrow \Theta(g(n)) = \Theta(n^3) \therefore O(n^3)$$

By taking off (-i+1) from each iteration in the original summation, then it will be a bigger. j covers more iterations than j = i+1 does therefore making it bigger. i is always going to be a positive number so not subtracting from it every iteration will always give you a bigger number.

(c) Although Dumbledores algorithm is a natural way to solve the problemafter all, it just iterates through the relevant elements of B, filling in a value for each it contains some highly unnecessary sources of inefficiency. Give an algorithm that solves this problem in time O(g(n)/n) (asymptotically faster) and prove its correctness.

```
dumbledoreSolve(A){
     for i = 1 to n{
         for j = 1 to n{
             if (j = i)
                 B[i,j] = A[i]
             else
                 B[i,j] = B[i, j-1] + A[j]

        }}}
```

$$O(\frac{g(n)}{n}) = O(\frac{n^3}{n}) = O(n^2)$$

$\sum_{i=1}^{n}(\sum_{j=i+1}^{n} O(1)) = \frac{1}{2}n(n+1) \Rightarrow \Theta(n^2) < n^3$ therefore faster.

Proof:
Base Case:
i = 1 , j = 1. B[i,j] = A[i] if j = i.
Induction step:
increase j, j = j+1 and reset i so j = i or increase i, i = i + 1 and reset j so i =j.
Terminates when i = n.