

1. (10 points) 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 (you can refer to the moodle lecture notes).
 - (a) What is the asymptotic running time of QuickSort when every element of the input A is identical, i.e., for $1 \leq i, j \leq n$, $A[i] = A[j]$?
 - (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?
 - (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.

2. (20 points) Solve the following recurrence relations using any of the following methods: unrolling, substitution, or recurrence tree (include tree diagram). For each case, show your work.

(a) $T(n) = T(n - 2) + C$ if $n > 1$, and $T(n) = C$ otherwise

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

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

(d) $T(n) = T(\sqrt{n}) + 1$ if $n \geq 2$, and $T(n) = 0$ otherwise

3. (20 points) Use the Master Theorem to solve the following recurrence relations. For each recurrence, either give the asymptotic solution using the Master Theorem (state which case), or else state the Master Theorem doesn't apply.

(a) $T(n) = T(\frac{3n}{4}) + 2$

(b) $T(n) = 3T(\frac{n}{4}) + n \lg n$

(c) $T(n) = 8T(\frac{n}{3}) + 2^n$

(d) $T(n) = T(\frac{n}{2}) + T(\frac{n}{4}) + n^2$

(e) $T(n) = 100T(\frac{n}{42}) + \lg n$

4. (30 points) 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 glows the opposite color of what the other crystal ball is (i.e. If the other crystal ball is accurate, it will glow red. If the other crystal ball is inaccurate it will glow green). You quickly notice that there are two possible test outcomes:

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

- (a) Prove that if $n/2$ or more crystal balls are inaccurate, Professor Trelawney cannot necessarily determine which crystal balls are tainted using any strategy based on this kind of pairwise test.
- (b) Consider the problem of finding a single good crystal ball from among the n crystal balls, and suppose Professor Trelawney knows that more than $n/2$ of the crystal balls are accurate, but not which ones. Prove that $\lfloor n/2 \rfloor$ pairwise tests are sufficient to reduce the problem to one of nearly half the size.
- (c) Now, under the same assumptions as part (4b), 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.

5. (10 points) Harry needs your help breaking into a dwarven lock box. The lock box projects an array A consisting of n integers $A[1], A[2], \dots, A[n]$ and has you enter in a two-dimensional $n \times n$ array B – to open the box – in which $B[i, j]$ (for $i < j$) contains the sum of array elements $A[i]$ through $A[j]$, i.e., $B[i, j] = A[i] + A[i + 1] + \dots + A[j]$. (The value of array element $B[i, j]$ is left unspecified whenever $i \geq j$, so it doesn't matter what the output is for these values.)

Harry suggests the following simple algorithm to solve this problem:

```
dwarvenLockBox(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).
- (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.)

6. (20 points) Consider the following strategy for choosing a pivot element for the `Partition` subroutine of `QuickSort`, applied to an array A .
- Let n be the number of elements of the array A .
 - If $n \leq 24$, perform an Insertion Sort of A and return.
 - Otherwise:
 - Choose $2\lfloor n^{(1/2)} \rfloor$ elements at random from n ; let S be the new list with the chosen elements.
 - Sort the list S using Insertion Sort and use the median m of S as a pivot element.
 - Partition using m as a pivot.
 - Carry out `QuickSort` recursively on the two parts.
- (a) How much time does it take to sort S and find its median? Give a Θ bound.
- (b) If the element m obtained as the median of S is used as the pivot, what can we say about the sizes of the two partitions of the array A ?
- (c) Write a recurrence relation for the worst case running time of `QuickSort` with this pivoting strategy.