Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**
Problem Set 7b (47 points + 10 pts extra credit) **Fall 2019, CU-Boulder**

Advice 1: For every problem in this class, you must justify your answer: show how you arrived at it and why it is correct. If there are assumptions you need to make along the way, state those clearly.

Advice 2: Verbal reasoning is typically insufficient for full credit. Instead, write a logical argument, in the style of a mathematical proof.

Instructions for submitting your solution:

- The solutions **should be typed** and we cannot accept hand-written solutions. Here's a short intro to Latex.
- You should submit your work through **Gradescope** only.
- If you don't have an account on it, sign up for one using your CU email. You should have gotten an email to sign up. If your name based CU email doesn't work, try the identikey@colorado.edu version.
- Gradescope will only accept **.pdf** files (except for code files that should be submitted separately on Gradescope if a problem set has them) and **try to fit your work in the box provided**.
- You cannot submit a pdf which has less pages than what we provided you as Gradescope won't allow it.

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**

Problem Set 7b (47 points + 10 pts extra credit) **Fall 2019, CU-Boulder**

Important: This assignment has two (Q2, Q3) coding questions.

- You need to submit two python files, one for each question.
- The .py file should run for you to get points and name the file as following -
  If Q2 asks for a python code, please submit it with the following naming convention - `Lastname-Firstname-PS7b-Q2.py`.
- You need to submit the code via Canvas but the table/plot/result should be on the main .pdf.

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**

Problem Set 7b (47 points + 10 pts extra credit) **Fall 2019, CU-Boulder**

1. (7 pts) Suppose that we modify the `Partition` algorithm in QuickSort in such a way that on alternating levels of the recursion tree, `Partition` either chooses the best possible pivot or the worst possible pivot.

   (a) (1 pt) What are the best possible and the worst possible pivots for Quicksort?
   Solution.

   The best possible pivot is when the pivot value is the median of the array if sorted, otherwise the average of the array.
   The worst possible pivot is when the pivot value is either a maximum or a minimum of the array.

   (b) (4 pts) Write down a recurrence relation for this version of QuickSort and give its asymptotic solution.
   Solution.

   if we know that the modification to the algorithm will cause a best case and worse case every other iteration then the new run time should be the average time of both best case / worst case run time.

$$T(n) = \frac{2T(n/2) + \theta(n) + T(n-1) + \theta(n)}{2}$$

$\rightarrow$

$$T(n) = \theta(n) * \frac{2T(n/2) + T(n-1)}{2}$$

$\rightarrow$

$$T(n) = \frac{1}{2} * (2T(n/2) + T(n-1)) * \theta(n)$$

   Runtime Complexity: $n^{log_{\frac{3}{2}}2}$

   (c) (2 pts) Provide a verbal explanation of how this `Partition` algorithm affects the running time of QuickSort.
   Solution. This modification makes the run time a linear function that is only affected by the size of the input because we know intuitively how long each iteration should take (either longest or shortest) as opposed to the time at every

iteration being random (random because we don't know how well of a pivot was chosen at that iteration).

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**

Problem Set 7b (47 points + 10 pts extra credit) **Fall 2019, CU-Boulder**

2. *(14 pts total) In PS1b, you were asked to count flips in a sorting algorithm with quadratic running time. The problem definition looked something like this:*
Let $A = \langle a_1, a_2, \ldots, a_n \rangle$ be an array of numbers. Let's define a 'flip' as a pair of distinct indices $i, j \in \{1, 2, \ldots, n\}$ such that $i < j$ but $a_i > a_j$. That is, a_i and a_j are out of order.
For example - In the array A = [1, 3, 5, 2, 4, 6], (3, 2), (5, 2) and (5, 4) are the only flips i.e. the total number of flips is 3. (Note that in this example the indices are the same as the actual values)

   (a) (14 pts) Write a Python program with the following features:

      i. (2 pts) Generates a sequence of n numbers in the range $[1, \ldots, n]$ and then randomly shuffles them.

      ii. (2 pts) Implements a $\theta(n^2)$ sorting routine that counts the number of flips in the array.

      iii. (5 pts) Implements a sorting routine with $\theta(nlgn)$ running time that counts the number of flips in the array. **Hint: Mergesort**

      iv. (5 pts) Run your code, both sorting algorithms, on values of n from $[2, 2^2, 2^3, ...2^{12}]$ and present your results in a table or labeled plot. Result with no supporting code will not get points.

   Follow the naming convention for python code mentioned on Page 2.

Solution.

$TABLEFORn^2algorithm$

Element Count	Flip Count
2	1
4	2
8	17
16	53
32	255
64	1059
128	4412
256	15551
512	68134
1024	258766
2048	1042113
4096	4228045

$TABLEFORnlognalgorithm$

Element Count	Flip Count
2	0
4	2
8	10
16	31
32	100
64	251
128	622
256	1472
512	3480
1024	7972
2048	17998
4096	39944

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**
Problem Set 7b (47 points + 10 pts extra credit) **Fall 2019, CU-Boulder**

3. (10 pts) Help the Mad Scientist calculate his h-index. According to Wikipedia: "A scientist has index h if h of their N papers have at least h citations each, and the other $N - h$ papers have no more than h citations each."

   For this question, write a Python program that calculates the h-index for a given input array. The array contains the number of citations for N papers, sorted in descending order (each citation is a non-negative integer). Your Python program needs to implement a divide and conquer algorithm that takes the *citations* array as input to outputs the h-index.

   Example:
   Input: citations = [6,5,3,1,0]
   Output: 3
   Explanation: [6,5,3,1,0] means the researcher has 5 papers with 6, 5, 3, 1, 0 citations respectively. Since the researcher has 3 papers with at least 3 citations each and the remaining two with no more than 3 citations each, the h-index is 3.
   Note: If there are several possible values for h, the maximum value is the h-index.
   Hint: Think how will you find it by a linear scan? You can then make your "search" more efficient.

   Do not submit anything on the .pdf for this question.
   Follow the naming convention for the python code mentioned on Page 2.

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**

Problem Set 7b (47 points + 10 pts extra credit) **Fall 2019, CU-Boulder**

4. (16 pts) Consider the following strategy for choosing a pivot element for the `Partition` subroutine of QuickSort, applied to an array A.

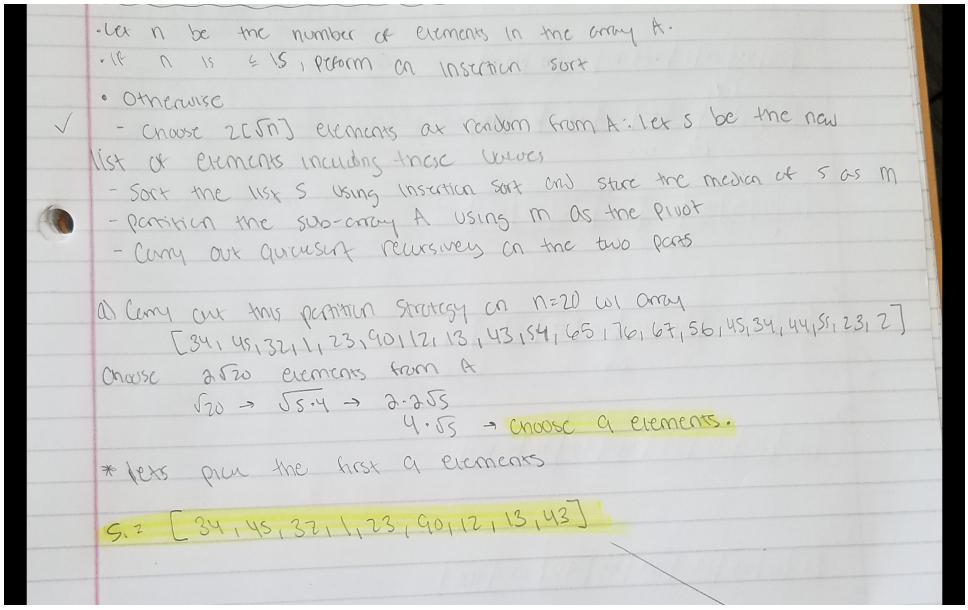
   - Let n be the number of elements of the array A.
   - If $n \leq 15$, perform an Insertion Sort of A and return.
   - Otherwise:
     - Choose $2\lfloor\sqrt{n}\rfloor$ elements at random from A; let S be the new list with the chosen elements.
     - Sort the list S using Insertion Sort and store the median of S as m.
     - Partition the sub-array of A using m as a pivot.
     - Carry out QuickSort recursively on the two parts.

   (a) (4 pts) Using the following array A with $n = 20$, show one iteration of this partitioning strategy on the array

   $$A = [34, 45, 32, 1, 23, 90, 12, 13, 43, 54, 65, 76, 67, 56, 45, 34, 44, 55, 23, 2]$$

   . Clearly identify all variables.

   Solution.

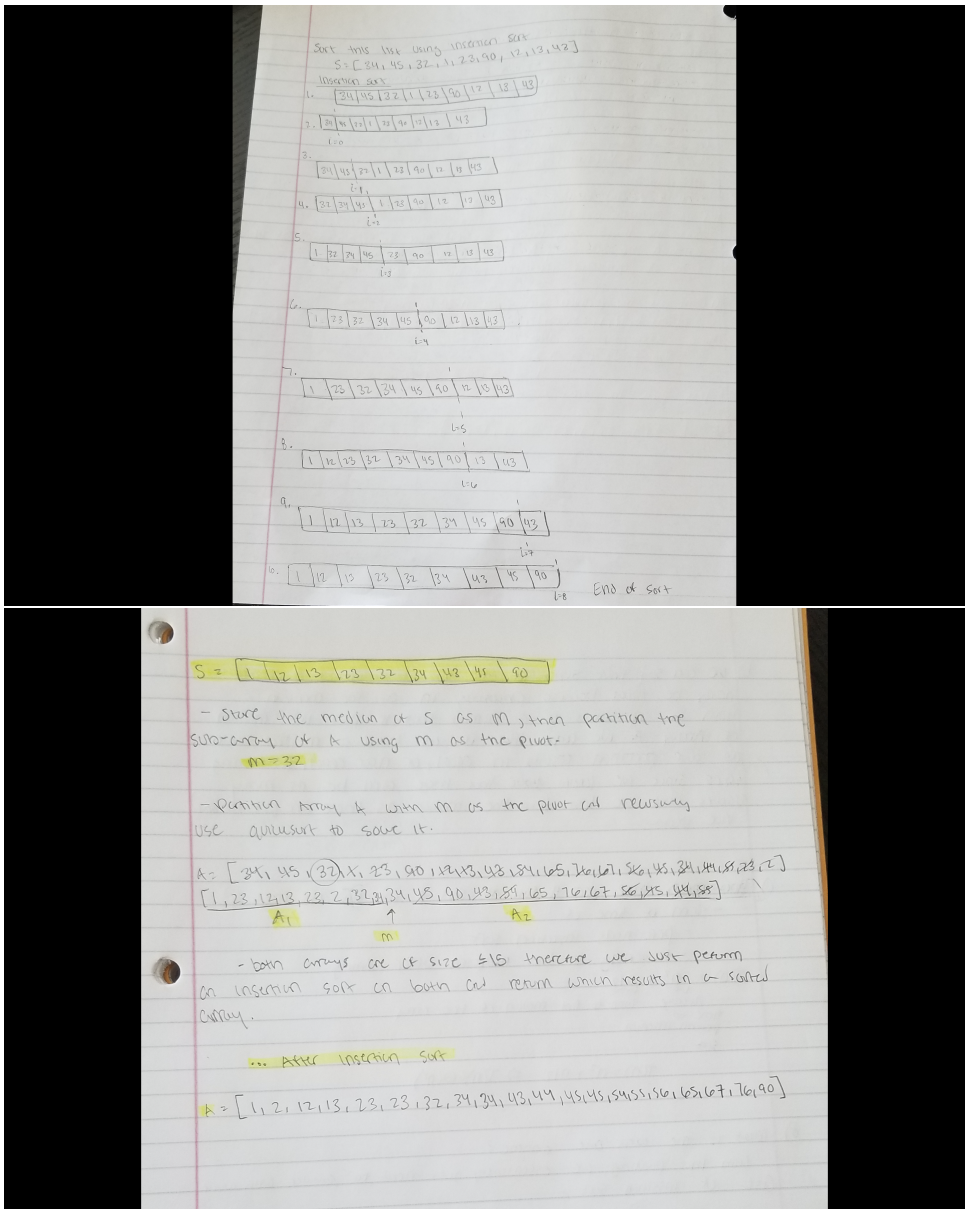


Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**

Problem Set 7b (47 points + 10 pts extra credit) **Fall 2019, CU-Boulder**



(b) (4 pts) 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? **Hint: Think about the best and worst possible selections for the values in S.**

Solution.

We can say that the sizes of the two sub arrays of A should nearly be equal because choosing 'm' as the median from 's' guarantees a number that is near the median of array A. We now that if we have a median value as our partition that it results in near equal partitioning since we that there will be as many values that are smaller than the pivot as there are values that are greater than the pivot and vice versa.

(c) (3 pts) How much time does it take to sort S and find its median? Give a Θ bound.

Solution.

should be the sum of the time to perform insertion sort and plus the time it takes to find the median

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

$\implies T(n) \in \Theta(n^2)$

(d) (5 pts) Write a recurrence relation for the worst case running time of QuickSort with this pivoting strategy.

Solution.

COST	TIME
c1	n^2 (time for insertion sort)
c2	$T(n/2)$(assuming best split given from Insertion Sort pivoting strategy)
c3	$T(n/2)$

$\implies T(n) + n^2 + 2T(\frac{n}{2})$

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**

Problem Set 7b (47 points + 10 pts extra credit) **Fall 2019, CU-Boulder**

5. (10 pts extra credit) Implement the bottles and lids algorithm that you wrote in assignment 7a and show that it functions correctly on randomly generated arrays representing 100 bottles and lids. Your algorithm needs to use a divide and conquer strategy to receive credit for this question.

Collaborated: Bao Nguyen, Zach Chommala