

CSCI 3104 PS1b

Jonathan Phouminh

TOTAL POINTS

31 / 44

QUESTION 1

34 pts

1.1 8 / 8

✓ **+ 8 pts Correct**

+ **4 pts** Correct python code

+ **4 pts** Correct Table/Graph

+ **0 pts** Incorrect Table

+ **2 pts** Code has only a single loop. The flips are counted only for flips that are consecutive.

+ **0 pts** Incorrect answer please go through the solutions provided.

+ **0 pts** Plagiarism

+ **4 pts** The python code is right.

+ **2 pts** The graph/Table should have had points until 2^{12}

+ **2 pts** Code is incorrect only consecutive flips are counted even though two loops are present

+ **0 pts** Please submit .py files and not .ipynb files

+ **3 pts** Partially correct python code

+ **3 pts** Partially correct Table

+ **2 pts** Generating numbers in the array only between 1-10. A size n array should have all the numbers from 1-n in shuffled order

+ **2 pts** python code loop partially correct

+ **0 pts** not attempted

+ **1 pts** python code to create the shuffled array

+ **0 pts** Code not submitted on canvas but table present

+ **0 pts** code has to be submitted in canvas. A table/Graph need to be submitted here. Unable to go through your code because special characters are missing.

1.2 2 / 4

+ **4 pts** Good work, Correct!!!

+ **0 pts** Empty or incomplete solution submitted.

Please refer to the solution file.

+ **3 pts** In the correct direction. But mention a value in terms of n by solving the mentioned formula completely. Please refer to the solution file.

+ **3.5 pts** In the correct direction. But evaluate the mentioned formula perfectly. Please refer to the solution file.

+ **1 pts** Incorrect assumption for greatest number of flips.

+ **0 pts** Incorrect assumption for greatest number of flips. Please refer to the solution file.

✓ **+ 2 pts Correct assumption but wrong or incomplete mathematical evaluation.**

+ **2 pts** Thinking in the right direction but more specific and detailed answer is expected

+ **2 pts** Thinking in the right direction but more detailed analysis needs to be done to arrive at the right answer

+ **3.5 pts** Incorrect formula

+ **1 pts** Thinking in the right direction but more mathematical answer is expected

💬 A small proof or justification is needed on how exactly you sum up terms. Please refer to the solution file.

1.3 10 / 10

✓ **+ 10 pts Correct**

+ **3 pts** Swap performed correctly

+ **3 pts** Correctly iterates through first loop and second loop does not give index out of bounds

+ **2 pts** Second loop iterates over all relevant elements

+ **2 pts** Indexes second loop by subtracting i and ignoring largest element (Conditional on rubric 4

being correct)

+ 0 pts Incorrect Solution

- 10 pts Plagiarism

1.4 0 / 4

+ 4 pts Good work!!!

+ 0 pts Empty or incomplete answer submitted.

Please refer the solution file.

+ 2 pts Good work, but your thought is focussing on the entire array's largest element index but the LI for inner loop focusses on making jth element the largest among first j elements.

+ 0 pts index i focusses on outer loop whereas LI is asked to be provided for the inner loop. Please read the question carefully and refer to the solution file.

+ 4 pts Good work but work on your explanation to be more precise. For reference, check the solution file.

+ 3 pts In the correct direction, but your solution is restrained to only two elements at a time. Please check solution file for reference.

+ 1 pts Your answer states that right most j elements are sorted. But here the LI is for inner loop and in that loop the jth element is max of all first j elements. Please check solution file for reference

+ 3 pts Please provide a more clear explanation using a variable like i or j. But your solution is in the correct direction

+ 0 pts Your explanation is unclear as it is not evident which loop you are talking about (inner loop or outer loop). Please refer solution file for reference.

+ 0 pts Your solution refers to last n-j elements but at every jth iteration, the A[j] is max of first j elements. Please refer solution file for reference.

+ 0 pts Inner loop isn't about min value in Array[i...n] but instead is about fixing max value at A[j] from A[0...j]

+ 0 pts Explained the meaning of for loop iteration range. But didn't explain LI

✓ + 0 pts [Click here to replace this description.](#)

+ 2 pts [Click here to replace this description.](#)

+ 1 pts [Click here to replace this description.](#)

+ 3 pts [Click here to replace this description.](#)

+ 0 pts Plagiarism

💬 LI is not an index (i according to your explanation). It is also not about comparing inner loop index j with outer loop index i. Please refer to the solution file.

1.5 2 / 8

✓ - 1 pts "Loop Invariant : The last i elements are sorted" is missing

✓ - 1 pts "Loop Invariant : the last i elements are not smaller than the first n-i elements" is missing

- 1 pts "Initialization : before the loop it is an empty set, Loop invariant hold trivial" is missing

- 1 pts Termination requires a more thorough explanation of how the final loop terminates

- 2 pts The assumption part of the Maintenance is not clear

- 2 pts Maintenance : The loop invariant confirmation of the next iteration is not clear.

- 2 pts In the Maintenance loop invariant "The last i elements are sorted" is missing

- 2 pts In the Maintenance loop invariant "the last i elements are not smaller than the first n-i elements" is missing

- 0 pts Good Work!

- 6 pts I cannot provide more points when your loop invariant is totally wrong. Please be more careful with indexes next time

- 8 pts Not correct

- 8 pts Potential Copy

- 8 pts Plagiarism

- 4 Point adjustment

💬 What I will say about your loop invariant is that it is a nice property but it really cannot tell anything about the progress of the loop. So the property only tells us that the array is fully sorted when j equals to zero. I do not want to discourage you because you seem like a student who can think out of the box. What I would suggest is to go to the internet and find

many examples of loop invariant and see the pattern between them. Good luck.

QUESTION 2

2 6 / 6

✓ + 6 pts **Correct**

+ 6 pts **Correct.** Even though the question did not specify to include the base case, it is good practice in an inductive proof to include them.

+ 4 pts **Missing summation notation and proof is nearly correct**

+ 4 pts **Missing some key proof steps**

+ 3 pts **Should deal with general case, not for specific value of k/r**

+ 2 pts **Wrong understanding of induction. k-case should imply k+1 case.**

+ 2 pts **Incorrect/Missing proof for inductive step**

+ 1 pts **Missing summation notation and proof is incorrect**

+ 0 pts **Not try or totally incorrect**

- 6 pts **Plagiarism**

QUESTION 3

3 3 / 4

+ 0 pts **Good answer! Explained very well.**

+ 0 pts **Not attempted or Incorrect answer. Please go through the solution provided.**

+ 1 pts **Partially correct. Please read and understand the concept again. Go through the solution provided.**

+ 1 pts **Initialization step is correct**

✓ + 0.5 pts **Initialization step is not explained well or it is not precise**

+ 2 pts **Maintenance step is correct.**

✓ + 1.5 pts **Maintenance step is in the right direction but has one mistake. When $A[i] \neq n$ in maintenance step, ret is not necessarily equal to -1, instead it may store the index of n if n is already found in $A[0 \dots i-1]$.**

+ 0 pts **Maintenance step is incorrect**

+ 0.5 pts **Maintenance step is not explained well**

✓ + 1 pts **Termination step is correct**

+ 0.5 pts **Termination step is not explained**

precisely

+ 2 pts **Please explain the answer properly with all steps**

+ 0 pts **Initialization step is not present or incorrect**

+ 0 pts **Termination step is incorrect**

+ 1 pts **Maintenance step explanation is in the right direction but not explained clearly**

- 4 pts **Plagiarism**

💬 **The initialization step is incorrect. Please read the concept again**

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms
Problem Set 1b (44 points)

Profs. Hoenigman & Agrawal
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.

-
1. (34 pts total) Let $A = \langle a_1, a_2, \dots, a_n \rangle$ be an array of numbers. Let's define a 'flip' as a pair of distinct indices $i, j \in \{1, 2, \dots, n\}$ 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) (8 pts) Write a Python code for an algorithm, which takes as input a positive integer n , **randomly shuffles an array of size n** with elements $[1, \dots, n]$ and counts the total number of flips in the shuffled array.

Also, run your code on a bunch of n values from $[2, 2^2, 2^3, \dots, 2^{20}]$ and present your result in a table with one column as the value of n and another as the number of flips. Alternatively, you can present your table in form of a labeled plot with the

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms
Problem Set 1b (44 points)

Profs. Hoenigman & Agrawal
Fall 2019, CU-Boulder

2 columns forming the 2 axes.

Note: The .py file should run for you to get points and name the file as `Lastname-Firstname-MMDD-PSXi.pdf`. You need to submit the code via Canvas but the table or plot should be on the main .pdf.

Element Count	Flip Count
2	1
4	5
8	15
16	70
32	207
64	1136
128	4189
256	15986
512	63060
1024	259556
2048	1055185
4096	4188112

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms
Problem Set 1b (44 points)

Profs. Hoenigman & Agrawal
Fall 2019, CU-Boulder

- (b) (4 pts) At most, how many flips can A contain in terms of the array size n ? Hint: The code you wrote in (a) can help you find this. Explain your answer with a short statement.

At most the number of total flips of array size n will be equal to the sum of each sub-array from $\text{array}[k:n]$ for each k , which can be written as

$$\sum_{k=1}^n n - 1$$

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms
Problem Set 1b (44 points)

Profs. Hoenigman & Agrawal
Fall 2019, CU-Boulder

- (c) (10 pts) We say that A is sorted if A has no flips. Design a sorting algorithm that, on each pass through A , examines each pair of consecutive elements. If a consecutive pair forms a flip, the algorithm swaps the elements (to fix the out of order pair). So, if your array A was $[4,2,7,3,6,9,10]$, your first pass should swap 4 and 2, then compare (but not swap) 4 and 7, then swap 7 and 3, then swap 7 and 6, etc. Formulate pseudo-code for this algorithm, using nested for loops.

Hint: After the first pass of the outer loop think about where the largest element would be. The second pass can then safely ignore the largest element because it's already in its desired location. You should keep repeating the process for all elements not in their desired spot.

```
def flipsort(array):  
    initiate  $i = 0$  and  $j = \text{length of array} - 1$   
    have a condition that tests if array is empty which we know when  $j = -1$   
    while loop runs as long as  $j > i$ , essentially as long as  $j$  is not equal to zero
```

Inside while loop will be an inner for loop that will range from i to j that will check the sub-array from $\text{array}[0:j]$. Inside this loop have a condition that checks if $\text{array}[i]$ and $\text{array}[i+1]$ are a flip, if this is indeed a flip, swap them.

after the inner loop finishes executing, decrement the value of j since the max element of the sub-array iteration that just finished will be in the proper spot

Reiterate over sub-array $[0:j-1]$ until outer loop condition is met and terminates.

At termination of outer loop, the array will be sorted.

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms
Problem Set 1b (44 points)

Profs. Hoenigman & Agrawal
Fall 2019, CU-Boulder

Another way to view. . .

def flipsort(array):

i = 0

j = len(array)-1

conditional that checks if array is empty

outer loop: while j greater i

inner loop: for i in range (j) , loop that checks sub-array up until j

inner loop condition that checks if array[i] and array[i+1] are a flip

if that inner loop conditional is true then swap

At the exit of the inner loop ,

$$j = j - 1$$

since the max element of current iteration will be in the proper spot.

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms
Problem Set 1b (44 points)

Profs. Hoenigman & Agrawal
Fall 2019, CU-Boulder

- (d) (4 pts) Your algorithm has an inner loop and an outer loop. Provide the 'useful' loop invariant (LI) for the inner loop. You don't need to show the complete LI proof.

In the second loop the useful loop invariant is i because i will either be less than j , telling us that all potential comparisons and swaps have not yet been seen or if i equals j then all potential swaps and comparison have been seen.

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms
Problem Set 1b (44 points)

Profs. Hoenigman & Agrawal
Fall 2019, CU-Boulder

- (e) (8 pts) Assume that the inner loop works correctly. Using a loop-invariant proof for the outer loop, formally prove that your pseudo-code correctly sorts the given array. Be sure that your loop invariant and proof cover the initialization, maintenance, and termination conditions.

LI: The loop invariant is variable j possessing multiple properties such that j is always equal to either 0, telling us that the array is sorted or, j is greater than zero, telling us that the array has not been fully sorted

Initiation: Loop is initiated at $i = 0$ and $j = \text{length of array} - 1$ meaning that the entire array has not been checked for all potential flips and swaps, therefore array is not yet sorted and our LI holds true

Maintenance: If at the i th iteration when j is still greater than 0, the sub-array from $[j:\text{length of array} - 1]$ will be sorted but the sub-array from $[0:j]$ will not be sorted and more swaps are needed, and our loop invariant holds true. If at the start of the i th iteration when j is equal to zero, the loop will trigger a termination and the array from $[0:\text{length of array} - 1]$ will be sorted, and our LI holds true.

Termination: At termination, when j equals zero, the given array will be sorted from least to greatest which is what our algorithm was intended for and therefore it works.

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms
Problem Set 1b (44 points)

Prof. Hoenigman & Agrawal
Fall 2019, CU-Boulder

2. (6 pt) If r is a real number not equal to 1, then for every $n \geq 0$,

$$\sum_{i=0}^n r^i = \frac{(1 - r^{n+1})}{(1 - r)}.$$

Rewrite the inductive hypothesis from Q3 on PS1a and provide the inductive step to complete the proof by induction. You can refer to Q3 on PS1a to recollect the first 2 steps.

Base Case, let $n = 0$

$$\sum_{i=0}^0 r^i = \frac{(1 - r^{0+1})}{(1 - r)} 1 = \frac{(1 - r)}{(1 - r)} 1 = 1$$

Inductive Hypothesis Assume,

$$\sum_{i=0}^k r^i = \frac{(1 - r^{k+1})}{(1 - r)}.$$

show $n = k + 1$ holds

$$\sum_{i=0}^k r^i + r^{k+1} = \frac{(1 - r^{k+1})}{(1 - r)} + r^{k+1}.$$

$$\begin{aligned} & \frac{(1 - r^{k+1})}{(1 - r)} + \frac{r^{k+1}}{1} \\ & \frac{(1 - r^{k+1})}{(1 - r)} + \frac{r^{k+1}}{(1 - r)} \\ & \frac{(1 - r^{k+1})}{(1 - r)} + \frac{r^{k+1} - r^{k+2}}{(1 - r)} \\ & \frac{1 - r^{k+2}}{1 - r} \end{aligned}$$

By Principle of Math Induction the original equation holds for all values, n , greater than or equal to zero. QED

Name: Jonathan Phouminh

ID: 106054641

CSCI 3104, Algorithms
Problem Set 1b (44 points)

Profs. Hoenigman & Agrawal
Fall 2019, CU-Boulder

3. (4 pt) Refer to Q2b on PS1a and finish the LI based proof with all the steps.

Loop Invariant: At the start of the i th iteration, ret has the index of n if n exists in $a[0 \dots i-1]$ otherwise $ret = -1$

Initialization: loop initialized at $i = 0$ thus searching the sub-array at $a[0]$, leaving 2 cases.

case 1) $a[0]$ is not equal to n , leaving ret to be equal to -1 , LI holds true

case 2) $a[0]$ is the element n and ret will equal i , LI holds true

Maintenance: For the i th iteration there will be two cases

case 1) if $a[i]$ is equal to n , ret will equal i

case 2) if $a[i]$ is not equal to n , ret will still be equal to -1

loop invariant holds true in both cases

Termination: Loop terminates when i is equal to $n-1$, ret will be equal to the index of the found element n or -1 if not found, which is what we wanted our algorithm to do. Therefore our algorithm is correct.