CS 3510 A Midterm Exam

Hyunwoo Kim

TOTAL POINTS

83 / 187

QUESTION 1

1 **10 / 19**

- **0 pts** Correct

✓ **- 9 pts** *Incorrect, but showed relevant work*

- **19 pts** Incorrect

- **8 pts** Correct, but didn't show relevant work.

QUESTION 2

28 pts

2.1 **14 / 14**

✓ **- 0 pts** *Correct*

- **3 pts** For any small error with indices, start, end or anything else which is not a part of the core algorithm for this problem.

- **7 pts** If the observation is not stated

- **7 pts** If the observation is stated, but binary search algorithm is not described/incorrectly described.

- **10 pts** If complexity is not $$O(logn)$$

- **14 pts** Completely incorrect or missing

- **14 pts** Writing illegible

2.2 **7 / 7**

✓ **- 0 pts** *Correct*

- **5 pts** Incorrect by some justification

- **7 pts** No justification

- **7 pts** Writing illegible

2.3 **7 / 7**

✓ **- 0 pts** *Correct*

- **3 pts** Correct complexity with insufficient/incorrect justification

- **5 pts** Incorrect complexity but with some justification

- **7 pts** Incorrect complexity and no justification

- **7 pts** Writing illegible

QUESTION 3

28 pts

3.1 **10 / 14**

- **0 pts** Correct

✓ **- 4 pts** *Does not search correctly*

- **5 pts** Does not ignore 1 edges

- **5 pts** Does not run in linear time

- **14 pts** Blank

3.2 **0 / 14**

- **0 pts** Correct

- **4 pts** Takes more than linear time

- **5 pts** Does not correctly search 0 edges

- **5 pts** Does not correctly search 1 edges

✓ **- 14 pts** *Blank*

QUESTION 4

4 **21 / 28**

- **0 pts** Correct
- **7 pts** Algorithm returns first value of k for which F(k) = k that is found (may not be the smallest value)
- **14 pts** if binary search is suggested but range calculation is not correct or is not done in O(logn)
- **14 pts** If range is found correctly but binary search is not applied
- **21 pts** Complexity not $$O(logn)$$
- **28 pts** Missing/Blank/Writes code or pseudocode
- **28 pts** Illegible
- **28 pts** Incorrect

- 7 *Point adjustment*

Does not specify what to do if a number k returns F(k) = k

1 Perfect start! Keep going

QUESTION 5

5 **0 / 28**

- **0 pts** Correct

✓ **- 7 pts** *No explanation for meaning of $$(k - |S|)/(n - i + 1)$$*

✓ **- 7 pts** *Does not explain why choosing $$i$$ w.p. $$(k - |S|)/(n - i + 1)$$ is correct*

✓ **- 14 pts** *Does not mention generating subsets w/ equal probability/adding each element w/ equal probability*

- **28 pts** Blank/Illegible

QUESTION 6

6 **0 / 28**

- **0 pts** Correct
- **7 pts** Does not search for a 400 digit number
- **7 pts** Does not randomly generate a 400 digit number
- **14 pts** Does not find a prime or test for primality
- **14 pts** Running time exponential in the number of bits
- **7 pts** Conducts a brute force search of the number space

✓ **- 28 pts** *Blank*

- **28 pts** Illegible

QUESTION 7

7 **14 / 28**

- **0 pts** Correct

✓ **- 7 pts** *Does not compare $$\log\log(n)$$ with $$\log(n)$$*

- **7 pts** Makes assumptions about the data structure

✓ **- 7 pts** *Does not explain why sorting bound is violated*

- **7 pts** Does not mention lower bound for sorting
- **28 pts** Blank/illegible

CS 3510: Design & Analysis of Algorithms (Section A)

Midterm Exam

Professor Merrick Furst **March 9, 2023**

- **You have 75 minutes to complete this exam.**
- **The midterm is out of 187 points.**
- **Your solutions must be in plain English and with mathematical expressions.**
- **Unless otherwise stated, the fastest (and correct) algorithms will receive more credit. If we ask for a specific running time, a correct solution achieving it will receive full credit even if a faster solution exists.**
- **Do not use pseudo-code. Your answer will receive zero credit even if the pseudo-code is correct.**

Name:	Hyunwoo Kim
GTID:	903561085
GT Username:	hkim3019

Name: Hyunwoo Kim GTID: 903561085

1. (19 points) Let $f(n) = \sum_{i=1}^{n} \frac{1}{i}$ and $g(n) = log(n)$. Which of the following statements is true?

[✓ **A.** $f(n) = \mathcal{O}(g(n))$]

○ **B.** $g(n) = \mathcal{O}(f(n))$

○ **C.** $f(n) = \mathcal{O}(g(n))$ **and** $g(n) = \mathcal{O}(f(n))$

○ **D.** None of the above.

$1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots$

which grows faster

$\boxed{\log(n)}$ or $\sum_{i=1}^{n} \frac{1}{i}$

7
5, 6, $\boxed{7}$ 8, 9
5, 6, 7, 8, 9, 10

Name: Hyunwoo Kim GTID: 903561085

5 10

2. (28 points) Let $A[0, ..., n-1]$ be a sorted array of n distinct elements from the $n+1$ element set $\{0, 1, 2, 3, 4, ..., n\}$.

(a) Design a divide and conquer algorithm that finds the missing element in $\mathcal{O}(log(n))$ time. 5

(b) Concisely justify the correctness of your algorithm.

(c) Analyze the running time of your algorithm.

ex) 0, 2, 3, 4, $\boxed{5}\boxed{6}\boxed{7}$ 8, 9
0, 1, 2, 3, $\boxed{4}$ 5, 6, 7, 8, 9

a)

1. We can solve this using a binary search algorithm. We set the lower bounds to 0 and the upper bounds to n-1. We find the middle index doing mid = (low + high) // 2. Then we compare A[mid] to mid.

2. If A[mid] == mid, it means every element to the left of mid is present, so we look on the right array from A[mid+1 ; n-1]. If A[mid] > mid, then an element is missing from the left array A[0 ; mid]. A[mid] cannot be smaller than mid as if the missing element is in the right of mid A[mid] == mid and if it's in the left A[mid] > mid.
If A[mid] == mid, then low = mid+1 / If A[mid] > mid, then high = mid

3. When we have one element left, we have 2 cases.
If A[mid] == mid, then return mid-1.
If A[mid] > mid, then return mid.

Name: Hyunwoo Kim GTID: 903561085

3. (28 points) You are given a graph $G = (V, E)$. Each edge in the graph has a cost associated with it which can be either 0 or 1. You are given a vertex s as the start node and a vertex k as the destination node.

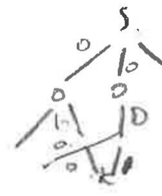
(a) (14 points) Design an algorithm that finds whether a zero-cost path exists from s to k in linear time. You just need to check if such a path exists or not; there is no need to return the path. Justify the correctness and time complexity of the algorithm.

(b) (14 points) Design an algorithm to find if a path exists from s to k such that the path has no 0 cost edges after a 1 cost edge has been used. For example:

Valid edge path: $0 \rightarrow 0 \rightarrow 0 \rightarrow 1 \rightarrow 1$

Invalid edge path: $0 \rightarrow 0 \rightarrow 1 \rightarrow 0$

This is invalid because a 0 cost edge is included after a 1 cost edge.



a) We start from s and perform BFS to explore other nodes. We ignore the edges if their cost is 1, we mark the edges taken as such and only go to unvisited nodes. If we get to k, we return true. If every node connected is marked as visited, and we haven't found k, we return false.

This algorithm works because

b)

Name: ______________________________ GTID: ______________________________

Name: Hyunwoo Kim GTID: 903561085

4. (28 points) Let F be a function defined on the natural numbers. $F(k)$ returns k if $n \leq k \leq 2n$, returns $-\infty$ if $k < n$ and returns $+\infty$ if $k > 2n$ (all in constant time) for some positive integer n. You are not given the value of n. Design an algorithm that determines the value of n in $\mathcal{O}(log(n))$ time.

We run $F(s)$ on a random natural number s

If $F(s)$ returns $-\infty$, we know that

$s < n$, so we set $s = 2s$.

If $F(s)$ returns ∞, we know that

$s > 2n$, so we set $s = s // 2$

If $F(s)$ returns s, then we know

$n \leq s \leq 2n$, so we

1

Name: ______________________ GTID: ______________________

5. (28 points) Given an integer k, you want to create a random subset of size k of the elements in range $[1, 2, ..., n]$. Assume $k < n$. Does the algorithm below work? Explain why or why not. (Hint: Remember the argument about randomizing an array in class.)

Algorithm 1 Randomized-Algorithm (k, n):

```
S := empty set (the subset to be created)
for i := 1 to n do
    r := randomly chosen number in [0, 1]
    if $r \leq (k - |S|) / (n - i + 1)$ then
        Add i to S
```

The algorithm doesn't work.

Name: ______________________ GTID: ______________________

6. (28 points) Let $\pi(n)$ be the number of prime numbers less than or equal to n. $\pi(n)$ is known to be close to $\frac{n}{ln(n)}$. Knowing this, how would you go about finding a 400 ~~bit~~ digit prime number?

$n \approx \ln(n) \cdot \pi(n)$

Name: Hyunroo Kim GTID: 903561085

7. (28 points) Suppose we want to design a data structure that stores comparable objects. We want it to support two operations.

1. Add an object in $\mathcal{O}(1)$ time.
2. Remove the object with smallest value in $\mathcal{O}(\log(\log(n))$ time where n is the number of elements in the data structure.

Explain why can't there be such a data structure.

For us to be able to add an object in O(1) time, we can't sort elements as we add. So the data structure is an unordered list. Now we want to find the object with the smallest value in O(log(logn)) time from an unsorted list. The fastest way to find the smallest value is the k^{th} smallest element algorithm where we pick an element at random and use it as pivot to split the problem. The algorithm runs in O(n) time on average. So we can't find the smallest value in O(log(log(n)) time.

Name: HYunWoo Kim GTID: 903561085

(Rough Work)

(Cont)

2. (C)

Since we're dividing the problem size in half every time we set a new low or high, and all we're doing is checking if A[mid] == mid, which takes O(1) time, the running time is log(n) · 1 = O(log(n)).

(b) The algorithm relies on the fact that if we look at a single element in A, and compare it to its index, it's going to match up unless there's a missing element on or to the left of mid. We divide the problem in half using that until we get to the last element which is where there are two cases. If A[mid] == mid, that means we return mid + 1 as the element missing is from the right. If not, we return mid as mid would've been in A[mid]'s spot.

Name: Hyunwoo Kim GTID: 903561085

2.

(Rough Work)

mid = 6

0, 1, 2, 3, $\boxed{4}$, 5, $\boxed{6}$, 8, 9

0 1 2 3 $\boxed{4}$ 5 6 7 8 9

0, 2, 3, 4, $\boxed{5}$ 6, 7, 8, 9

0 1 2 3 4 9

e

== ⇒ R

> ⇒ Left

< ⇒ Right

0, 1,

0, 1, 2, 3, 4

0, 1, 2, 3, 4

0, 1, $\boxed{2}$, 3, 4

0, 1, $\boxed{2}$, 3, 4, 5

3, $\boxed{4}$

3, $\boxed{4}$, 5

$\boxed{0}$ $\boxed{2}$

0, $\boxed{1}$, 2

$\boxed{2}$

5

0, 2, $\boxed{3}$, 4

mid = 2

A[mid] = 3

0, 1, 2, 3, 4

equal ==

mid − 1

bigger

= mid

A[mid] > mid

$\boxed{0}$, $\boxed{2}$

0, $\boxed{1}$, 2, 3, 4

0, $\boxed{1}$ 2, 4

0, $\boxed{1}$ 2, 3, 4

0

0 1 2 3 4

$\boxed{2}$ 4

$\boxed{2}$, 3, 4

0, 1, $\boxed{2}$ 4

0, 1, 2, 3, 4

A[mid] > mid

$\boxed{4}$

3, 4

$\boxed{4}$

$\boxed{3}$, 4