Homework 2

CS6033 Design and Analysis of Algorithms I

Fall 2023

(Sec. B, Prof. Yi-Jen Chiang)

Due: Wed. 10/4 by 1pm
(submit online on NYU Brightspace; one submission per group)
Maximum Score: 100 points

Note: This assignment has 3 pages.

1. (20 points)

Define a sequence of numbers S_n (for integers $n \geq 0$) recursively as follows:

$$S_n = \begin{cases} 1 & \text{if } n = 0, \\ 2 & \text{if } n = 1, \\ 3 & \text{if } n = 2, \\ 4 & \text{if } n = 3, \\ S_{n-1} + S_{n-4} & \text{if } n \geq 4. \end{cases}$$

(a) Prove **by induction** that $\sum_{i=4}^{n} S_{i-4} = S_n - 4$ for all integers $n \geq 4$. No credit will be given for a different method of proof. **(10 points)**

(b) Prove **by induction** that $S_{n+4} \geq \hat{\phi}^n$ for all integers $n \geq 0$, where $\hat{\phi} \in (1, 2)$ is a root of the equation $x^4 - x^3 - 1 = 0$. No credit will be given for a different method of proof.
(**Remark:** Such $\hat{\phi}$ exists and here is an argument to show why: If we let $f(x) = x^4 - x^3 - 1$, we have $f(1) < 0$ and $f(2) > 0$, and thus the equation $f(x) = 0$ has a solution (root) in the range $(1, 2)$ and we call this root $\hat{\phi}$ — this is just a background information and you do not need to worry about this part.) **(10 points)**

2. (12 points)

What is wrong with the following purported proof that all horses have the same color?
Proof by induction on the number of horses:

Basis Step. There is only one horse. Then clearly all horses have the same color.

Induction Hypothesis. In any group of up to n horses, all horses have the same color.

Induction Step. Consider a group of $n+1$ horses. Discard one horse; by the induction hypothesis, all the remaining horses have the same color. Now put that horse back and discard another; again all the remaining horses have the same color. So all the horses have the same color as the ones that were not discarded either time, and so they all have the same color.

3. (10 points)

Consider the following sequence of items

10, 36, 25, 54, 37, 12, 75, 68, 42, 86, 72, 90.

Insert these items in the order above, into an initially empty implicit binary max-heap. Show the heap throughout the process but write down the heap as an answer after every 3 INSERT() operations. (Just hand-simulate the behavior of the implicit binary max-heap, and draw the resulting heap, both throughout the entire process and the ones after every 3 INSERT() operations as answers (note that the former is to show all your work so that partial credits can be given in case you make mistakes, while the latter (answers) are the ones to be graded). Draw them in the form of a **tree** rather than an array.) **(1 + 2 + 3 + 4 = 10 points)**

4. (6 points)
Consider the implicit binary max-heap below. Show the results of performing 2 consecutive EXTRACT_MAX() operations on it. For **each** EXTRACT_MAX() operation, show the item extracted together with the resulting heap.

```
            80
         /      \
       45         50
     /  \       /  \
    28    33    36  32
    /\    /\    /
   7  4 15 8 25
```

(Just hand-simulate the behavior of the implicit binary max-heap, and draw the resulting heap after every EXTRACT_MAX() operation. Draw them in the form of a **tree** rather than an array.)
(**Note: 3 points for each result.**)

5. (36 points)
In a max-heap we can find the maximum easily, and in a min-heap we can find the minimum easily. Your task here is to extend the ideas so that we can find **both** the maximum and minimum easily. The new heap, called the implicit binary **max-min heap**, is similar to the implicit binary max-heap (or min-heap) except that the **heap properties** are different, as follows:

(i) if a node u is at an **even** level (the root is at level 0, and the children of the root are at level 1, etc.), then $key(u)$ is the **maximum** among the keys of the subtree rooted at u; and

(ii) if a node u is at an **odd** level, then $key(u)$ is the **minimum** among the keys of the subtree rooted at u.

(a) Draw an example of the binary max-min heap with 10 items (in the form of a **tree**, rather than an array). **(5 points)**

For the rest of this question, let A be the array of the implicit binary max-min heap containing n items with $A[1]$ being the root.
(b) How do we find the maximum in the heap A? How do we find the minimum? For each, briefly describe your algorithm and its worst-case running time. **(1 + 2 + 1 = 4 points)**

(c) The operation INSERT(A, key) is to insert an item with key value key into the heap A. Give an algorithm to carry out INSERT(A, key), and analyze its running time. It should run in $O(\log n)$

worst-case time. **(7 points)**

(d) The operation HEAPIFY(A, i) is to update the heap A so that the subtree rooted at $A[i]$ satisfies the **heap properties** (i) and (ii), given that **each** of the left and right subtrees of $A[i]$ already satisfies the heap properties (i) and (ii). Give an algorithm to carry out HEAPIFY(A, i), and analyze its running time. It should run in $O(\log n)$ worst-case time. **(11 points)**

(e) Given an array A with n items stored in $A[1], \cdots, A[n]$, the operation BUILD_HEAP(A, n) is to re-order these items so that A becomes a valid implicit binary max-min heap storing such n items in $A[1], \cdots, A[n]$. Design and analyze an algorithm to carry out BUILD_HEAP(A, n) as efficiently as possible. **(4 points)**

(f) Discuss how to perform EXTRACT_MAX(A) and EXTRACT_MIN(A) (report and remove the maximum (resp. minimum) item from the heap A) as efficiently as possible. For each, you need to give an algorithm and analyze its running time. **($2 + 3 = 5$ points)**

6. (16 points)
Given an implicit binary min-heap A in an array (where the root $A[1]$ stores the minimum item), a real number x and a positive integer k, design and analyze an algorithm to decide whether the k-th smallest item in A is $\leq x$. Your algorithm should run in $O(k)$ worst-case time and use at most $O(k)$ extra space, independent of the size of A.
Note: Your algorithm does **not** need to report the k-th smallest item in A; only its relationship with x needs to be decided, to be able to answer "yes" or "no".