Homework Assignment 2

CSE33101 Intro to Algorithms (Spring 2022)

Due: 2022-05-06 11:59 pm

Handwrite your answer to the following questions in English, scan it, and submit it to BlackBoard. **Illegible answers will not be graded** (zero points).

Total 10 points

1. (2 points) Explain that the running time of the PARTITION procedure of quicksort on a subarray of size n is $\Theta(n)$.

2. (3 points) Chebyshev's inequality says that the probability that a random variable is more than k standard deviations away from the mean is less than $1/k^2$. For $N = 1{,}000{,}000$, use Chebyshev's inequality to bound the probability that the number of compares used by quicksort is more than 100 million. Hint: Quicksort uses $2N \ln N$ (ln is the natural logarithm) compares on the average case (mean) to sort N keys, and the standard deviation of the number of compares is $0.65N$.

3. (3 points) Assume there is an empty max-priority queue A with the following heap procedures.

```
HEAP-EXTRACT-MAX(A)
1  if A.heap-size < 1
2       error "heap underflow"
3  max = A[1]
4  A[1] = A[A.heap-size]
5  A.heap-size = A.heap-size − 1
6  MAX-HEAPIFY(A, 1)
7  return max
```

```
MAX-HEAP-INSERT(A, key)
1  A.heap-size = A.heap-size + 1
2  A[A.heap-size] = −∞
3  HEAP-INCREASE-KEY(A, A.heap-size, key)
```

```
HEAP-INCREASE-KEY(A, i, key)
1  if key < A[i]
2       error "new key is smaller than current key"
3  A[i] = key
4  while i > 1 and A[PARENT(i)] < A[i]
5       exchange A[i] with A[PARENT(i)]
6       i = PARENT(i)
```

```
MAX-HEAPIFY(A, i)
1  l = LEFT(i)
2  r = RIGHT(i)
3  if l ≤ A.heap-size and A[l] > A[i]
4       largest = l
5  else largest = i
6  if r ≤ A.heap-size and A[r] > A[largest]
7       largest = r
8  if largest ≠ i
9       exchange A[i] with A[largest]
10      MAX-HEAPIFY(A, largest)
```

Illustrate (draw) the max-priority queue of each step of the following operations (draw total 12 priority queues). Assume that items are ordered in a reverse alphabetical manner in our max-priority queue.

- MAX-HEAP-INSERT(A, "P") → MAX-HEAP-INSERT(A, "Q") → MAX-HEAP-INSERT(A, "E") → HEAP-EXTRACT-MAX(A) → MAX-HEAP-INSERT(A, "X") → MAX-HEAP-INSERT(A, "A") → MAX-HEAP-INSERT(A, "M") → HEAP-EXTRACT-MAX(A) → MAX-HEAP-INSERT(A, "P") → MAX-HEAP-INSERT(A, "L") → MAX-HEAP-INSERT(A, "E") → HEAP-EXTRACT-MAX(A)

4. (2 points) Given an array A consisting of n positive integers, let's assume that we want to find a positive integer k such that the total sum of the distances between all elements of A and k becomes minimized, i.e., $sum = \min \sum_{i=1}^{n} d_i$, where $d_i = |A[i] - k|$ is the distance between $A[i]$ and k. Write **real code** (not pseudocode) that prints sum (not k), given an array A of length n. **Write (type) your code with a keyboard and upload it to the "comment section". Do not handwrite your code on the paper**.

   - Your code should be written either in **C or Python**. Your code should be compiled and run. It is all your responsibility to make sure that your code is error-free and has no typos.
   - **Type your code with a keyboard and upload it to the "comment section" in the "Assignment 2" menu. Do not include your code in your answer paper.**
   - Only the code uploaded to the comment section will be graded (not the code on your computer).
   - Do not use built-in libraries in C and Python.
   - Grading criteria:
     i. 2 points: pass all test cases (test cases are not provided)
     ii. 1 point: pass more than or equal to 30% and less than 100% of test cases
     iii. 0 point:
        1. If pass less than 30% of test cases or,
        2. If there are compile errors or,
        3. If code runs longer than 1 minute on the test cases.