MET CS 566 - Analysis of Algorithms
Assignment 3 - 20 Points

Please note the honor code policy regarding this Assignment. You may not discuss particular questions or discuss or transmit answers from the assignment with other people, except for the MET CS 566 teaching team.

Tasks

1. Max-Heapify (3 points): Illustrate the operation of MAX-HEAPIFY(A, 3) on the array

$A = (6, 8, 10, 9, 7, 5, 4, 13, 23, 1, 5, 7, 12, 4)$

Write your steps, and count the number of total swap operations (exchange of keys)

1	2	3	4	5	6	7	8	9	10	11	12	13	14
6	8	10	9	7	5	4	13	23	1	5	7	12	4

- The first row is the index
- The second row is the array value (In a heap, the key)

2. **Heap-Sort (3 points)**: Run the HEAP-Sort Algorithm on the above array and count up the total number of exchanges (You can assume the array has already been built as a heap)

- Write down the initial Max-heap array
- Write down the array after each key exchange in the HEAP-Sort.
- Write down the total number of key exchanges in the HEAP-Sort algorithm.
- No need to illustrate all your steps.

Note: Number of exchanges is the number of times that you swap the position of two keys (Values of the array) until it is a sorted array

3. **Heap-Median (4 points)**: Write pseudocode to extract the Median of an array using a heap structure. Name it Extract−Median(A) where A is an array. Develop an algorithm with the best running time.

- You can use similar pseudocode syntax similar to the CLRS book or lecture slides.
- Discuss what is the running time of your best algorithm

Hint: The efficient algorithm to extract median using heap needs to build two heaps: Max-heap to hold smaller half part of the values; Min-heap to save bigger half part of the values. The size difference between two heaps should be within 1. The key algorithm in Extract-Median is Median-Heapify (you may refer Max-Heapify algorithm in the lecture).

4. **Hashing by Chaining (3 points)**: Demonstrate what happens when we insert the keys $(6, 8, 10, 9, 7, 5, 4, 13, 23, 1, 5, 7, 12, 4)$ into a hash table with collisions resolved by chaining. Let the table have 9 slots, and let the hash function be $h(k) = k \bmod 9$

- Count the number of collisions
- How many collisions would the hash table have if you use $h(k) = k \bmod 7$?

Note: Hash table indexes start from zero.

5. Open Addressing - Linear Probing (3 points): Consider inserting the keys {31, 11, 5, 17, 25} into a hash table of length m = 7 using open addressing with the hash function

$$h(k,i) = (h'(k) + i) \, mod \, m$$
$$h'(k) = k \, mod \, 7$$

- Illustrate the result of inserting these keys using linear probing and provide insertion table index results
- Count the number of collisions

6. **Open Addressing - Double Hashing (4 points)**: Consider inserting the keys $\{31, 11, 5, 17, 25\}$ into a hash table of length $m = 7$ using open addressing with double Hashing

$$h(k,i) = (h_1(k) + i \times h_2(k)) \, mod \, m)$$
$$h_1(k) = k \, mod \, 7$$
$$h_2(k) = 1 + (k \, mod \, 3)$$

- Illustrate the result of inserting these keys using Double Hashing, and provide your resulting hash table.