Algorithm Analysis Homework 1

Due by 3/21(Fri.) through LMS

* Grading policy

Remember that correctness is an important criterion, but by no means the whole story. Grades on program will be based on:

1. Correct behavior on typical input: 70%
2. Adherence to specification: 10%
3. Correct behavior on extreme or unusual situations & reasonable recovery from unusual or incorrect inputs.: 10%
4. Readability ~ comments, Mnemonics identifier, clear program structure.: 10%

* Note

1. At header part of comment, list all the references you used when you do this homework.

   For ex)

   (1) 강의 slide chapter 6. page3-5

   (2) Blog: ** URL here **

   (3) book: "Algorithm analysis in C++" by Someone, chapter 5
2. You should not use the STL or any AI-powered services.

Write a C++ program that implements a max-priority queue using a heap data structure. (Note: If you do not use a heap, you will receive zero points for this homework. Additionally, element indices should start from 1, not 0.)

1) The maximum heap size is 30. If the heap reaches its maximum size, the program should display an error message and reject additional insertions.
2) Each element has three fields – student name, score, and class name.
3) The student name consists only of English letters and can have a maximum length of 20 characters.
   - You do not need to check the validity of the name.
   - We assume that the user always enters a valid name (i.e., containing only alphabetic characters and within the length limit).
4) The score is an integer ranging from 0 to 100 and serves as the key field.

   Case 1: The user always enters an integer, but it may be out of range (e.g., -1 or 101). You must validate the score and handle invalid inputs appropriately.

   Case 2 (optional): The user may enter non-numeric characters such as "1 3", "3o", or "1_3".

Your program should handle Case 1. If your program properly handles Case 2, you will receive an extra 20% points.

5) The student name and class name may contain spaces, such as "Tom S. Carpenter" or "Algorithm 101".

* Your program should include the following functions:

- INSERT(S, x): Inserts element x into the set S.
- MAXIMUM(S): Returns the element in S with the highest key.
- EXTRACT-MAX(S): Removes and returns the element in S with the highest key.
- INCREASE-KEY(S, x, k): Increases the key value of element x to k. If the new key value is smaller than the current one, the program should reject it and prompt the user to enter a new valid score.

You may implement additional functions as needed.

* When the program starts, it should repeatedly present a menu until the user enters 'Q' to quit.

Menu Options and Their Functionality:

I (Insert):

Prompts the user to enter a student name, score, and class name, then inserts the element into the priority queue. If the heap is full, the program should display an error message and reject the insertion.

D (Delete) (Extract-Max):

Removes and displays the record of the element with the largest score, then rebuilds the queue. If the queue is empty, the program should display "Cannot delete from an empty queue."

R (Retrieve Maximum):

Displays the name and key value of the element with the highest score (largest key). If the queue is empty, the program should display "Queue is empty."

N (Increase Key):

Prompts the user for an index in the queue and a new score for the corresponding element. Updates the key value of the element and repositions it accordingly within the heap. If the user enters a score lower than the current one, the program should reject it and prompt for a new valid score.

P (Print Queue):

Displays all elements currently in the queue in heap order (not sorted order).

Q (Quit): Terminates the program.

example)

```
*********** MENU ****************
I : Insert a new element into the queue.
D : Delete the element with the largest key from the queue.
R : Retrieve the element with the largest key.
N : Increase the key of an element in the queue.
P : Print all elements in the queue.
Q : Quit.

Choose menu: I
Enter the name of the student: David Bowie
Enter the score of the element: 80
Enter the class name: Algorithm 101
New element [David Bowie, 80, Algorithm 101] has been inserted.

*********** MENU ****************
I : Insert a new element into the queue.
D : Delete the element with the largest key from the queue.
R : Retrieve the element with the largest key.
N : Increase the key of an element in the queue.
P : Print all elements in the queue.
Q : Quit.

Choose menu: I
Enter the name of the student: Chris Martin
Enter the score of the element: 40
Enter the class name: Algorithm 102
New element [Chris Martin, 40, Algorithm 102] has been inserted.

*********** MENU ****************
I : Insert a new element into the queue.
D : Delete the element with the largest key from the queue.
R : Retrieve the element with the largest key.
N : Increase the key of an element in the queue.
P : Print all elements in the queue.
Q : Quit.

Choose menu: I
Enter the name of the student: Tom S. Carpenter
Enter the score of the element: 101
Invalid score. Please enter a valid integer between 0 and 100.
Enter the score of the element: 95
Enter the class name: Data Structures
New element [Tom S. Carpenter, 95, Data Structures] has been inserted.

*********** MENU ****************
I : Insert a new element into the queue.
D : Delete the element with the largest key from the queue.
R : Retrieve the element with the largest key.
N : Increase the key of an element in the queue.
P : Print all elements in the queue.
Q : Quit.

Choose menu: P
Current queue elements:
1. [Tom S. Carpenter, 95, Data Structures]
2. [David Bowie, 80, Algorithm 101]
3. [Chris Martin, 40, Algorithm 102]

*********** MENU ****************
I : Insert a new element into the queue.
D : Delete the element with the largest key from the queue.
R : Retrieve the element with the largest key.
N : Increase the key of an element in the queue.
```

```
P : Print all elements in the queue.
Q : Quit.

Choose menu: R
Element with the largest key: [Tom S. Carpenter, 95, Data Structures]

************ MENU *****************
I : Insert a new element into the queue.
D : Delete the element with the largest key from the queue.
R : Retrieve the element with the largest key.
N : Increase the key of an element in the queue.
P : Print all elements in the queue.
Q : Quit.

Choose menu: N
Enter the index of the element: 2
Enter the new score: 10
New score should be larger than current score. Please enter again.
Enter the new score: 85
Key updated. [David Bowie, 85, Algorithm 101] has been repositioned in the
 queue.

************ MENU *****************
I : Insert a new element into the queue.
D : Delete the element with the largest key from the queue.
R : Retrieve the element with the largest key.
N : Increase the key of an element in the queue.
P : Print all elements in the queue.
Q : Quit.

Choose menu: D
Deleted element: [Tom S. Carpenter, 95, Data Structures]

************ MENU *****************
I : Insert a new element into the queue.
D : Delete the element with the largest key from the queue.
R : Retrieve the element with the largest key.
N : Increase the key of an element in the queue.
P : Print all elements in the queue.
Q : Quit.

Choose menu: P
Current queue elements:
1. [David Bowie, 85, Algorithm 101]
2. [Chris Martin, 40, Algorithm 102]

************ MENU *****************
I : Insert a new element into the queue.
D : Delete the element with the largest key from the queue.
R : Retrieve the element with the largest key.
N : Increase the key of an element in the queue.
P : Print all elements in the queue.
Q : Quit.
Choose menu: Q
Program terminated.
```

Note:

1. Menu options are case-insensitive
   - Whether you enter 'i' or 'I', the program will recognize it as "Insert."
   - The same applies to other menu options (D, R, N, P, Q).

2. Score validation is strictly enforced
   - The score must be an integer between 0 and 100.
3. Names and class names maintain case sensitivity
   - "Alice Johnson" and "alice johnson" are treated as different names.
   - "Data Structures" and "data structures" are not considered identical.
4. Heap property is maintained after every insertion, deletion, or key update
   - The program ensures that the max-priority queue structure is always valid.
   - If a key is increased, the element is repositioned accordingly.
5. Program does not check for duplicate names
   - The same student name can appear multiple times with different scores or class names.
6. Indexing starts from 1
   - When increasing a key (N option), the user must enter an index starting from 1, not 0.
7. If R (Retrieve Maximum) is selected when the queue is empty, the program should display "Queue is empty."
8. If D (Delete / Extract-Max) is selected when the queue is empty, the program should display "Cannot delete from an empty queue."

7. (Optional): Does your program work correctly when there is a space before or after a menu choice, such as " P" or "p "? Which implementation is more reasonable: allowing spaces or restricting input to exact characters?