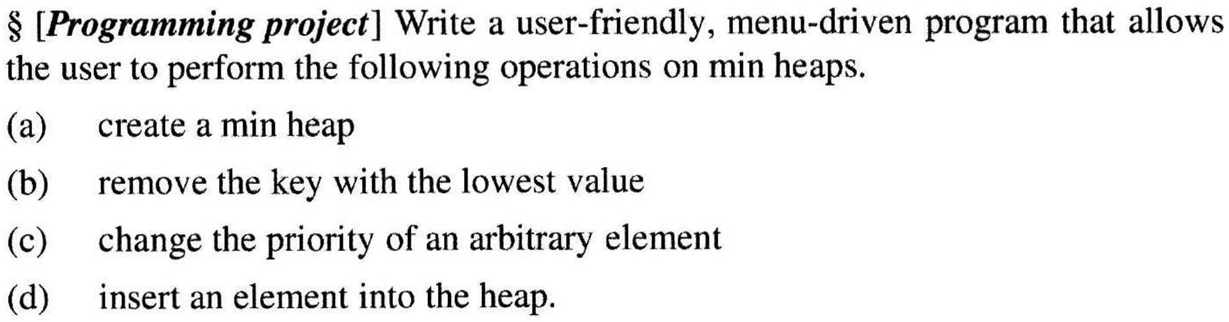
HW4

If your program takes more than a certain reasonable amount of time to finish, you get 0 points. Every program should be compilable and runnable in cspro. Every program should be written in C (at this time, No C++) and satisfy the requirements in the program specification. Copy check will be done.

1. Textbook (2nd Eds) p.230 #9.



§ [***Programming project***] Write a user-friendly, menu-driven program that allows the user to perform the following operations on min heaps.

(a) create a min heap

(b) remove the key with the lowest value

(c) change the priority of an arbitrary element

(d) insert an element into the heap.

Add the following operations:

(e) remove the selected priority

(f) search the heap if the heap has the item with an input key (priority).

Example output:

```
MIN Heap Operations

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:6
Remove Priority: 0
0 is not in the priority queue.

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:1
Enter a number: 11
n = 0
[1] = 11

1. Insert, 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority
0. Quit:1
Enter a number: 5
n = 1
[1] = 5
[2] = 11

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:1
Enter a number: 7
n = 2
[1] = 5
[2] = 11
[3] = 7
```

```
1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:1
Enter a number: 9
n = 3
[1] = 5
[2] = 9
[3] = 7
[4] = 11

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:6
Remove Priority: 8
8 is not in the priority queue.
[1] = 5
[2] = 9
[3] = 7
[4] = 11

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:6
Remove Priority: 11
[1] = 5
[2] = 9
[3] = 7

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:2
5 was deleted from the heap.
[1] = 7
[2] = 9

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:1
Enter a number: 16
n = 2
[1] = 7
[2] = 9
[3] = 16

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:1
Enter a number: 25
n = 3
[1] = 7
[2] = 9
[3] = 16
[4] = 25

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:4
The top priority is: 7.

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:5
Change Priority of: 16
New Priority: 40
[1] = 7
[2] = 9
[3] = 40
[4] = 25

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:5
Change Priority of: 40
New Priority: 1
[1] = 1
[2] = 9
[3] = 7
[4] = 25
```

```
1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:6
Remove Priority: 7
[1] = 1
[2] = 9
[3] = 25

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:3
Search for y: 0
0 is not in the heap.

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:3
Search for y: 9
9 was FOUND in position 2.

1. Insert 2. Delete 3. Search 4. Top Priority 5. Change Priority 6. Remove Priority 0.
Quit:0
```

2.Write a complete menu-driven program that allows a user to perform the following operations by modifying/using Program 5.15, 5.17:

(a) create a binary search tree

(b) search the current tree with an input key

(c) insert an element with a given input key to the current binary search tree

You should print the current state of the tree after inserting each element, using preorder traversal: root → left child → right child.

Example output:

```
Binary Search Tree Operations

1. Insert 2: Search 0. Quit:1
Enter a number: 6
[1] = 6

1. Insert 2: Search 0. Quit:1
Enter a number: 7
[1] = 6
[2] = 7

1. Insert 2: Search 0. Quit:1
Enter a number: 4
[1] = 6
[2] = 4
[3] = 7

1. Insert 2: Search 0. Quit:1
Enter a number: 2
[1] = 6
[2] = 4
[3] = 2
[4] = 7

1. Insert 2: Search 0. Quit:1
Enter a number: 5
```

```
[1] = 6
[2] = 4
[3] = 2
[4] = 5
[5] = 7

1. Insert 2: Search 0. Quit:2
Search for y: 6
6 was FOUND in the tree.

1. Insert 2: Search 0. Quit:2
Search for y: 8
8 is not in the tree.

1. Insert 2: Search 0. Quit:0
```