CS302: Data Structures using C++

Fall 2020

Final Exam Preparation - Additional Exercises: **Heap ADT using STL**

Kostas Alexis

Problem Description: Utilize STL for heaps in order to:

- Make a heap consisting of 100 random integers.
- Add a new value that is the mean of the random values you created in the previous step. Floor the value if needed.
- Delete the maximum element of the heap and
- Sort the heap.

Deliver code, and a *.txt file with the output of the terminal.

Solution: The solution is organized in three files, namely a) main.cpp, b) heapmean.h, and c) randgen.h.

File: main.cpp

```cpp
#include <iostream>
#include <algorithm>

#include "randgen.h"
#include "heapmean.h"

using namespace std;

int main(){

        vector<int> heap1;


        randGen(heap1);

        cout << "Our Vector of 100 random integers" << endl;

        for(auto i = heap1.begin() ; i != heap1.end(); ++i){

            cout << *i << " ";
        }

        make_heap(heap1.begin(), heap1.end());

        cout << endl << endl << "The Original Heap" << endl << endl;

        for(auto i = heap1.begin() ; i != heap1.end(); ++i){

            cout << *i << " ";
        }

        int mean = heapmean(heap1);


        cout << endl <<endl << "We are adding a new integer of " << mean << " " <
< endl;

        heap1.push_back(mean);
        push_heap(heap1.begin(), heap1.end());

        cout << endl << "Max of Heap is being deleted" << endl;


        pop_heap(heap1.begin(), heap1.end());
        heap1.pop_back();

        cout << endl << "Sorting the heap" << endl;


        make_heap(heap1.begin(), heap1.end());

        for(auto i = heap1.begin() ; i != heap1.end(); ++i){

            cout << *i << " ";
        }

        cout << endl;

    return 0;

}
```

File: heapmean.h

```
#ifndef HEAPMEAN
#define HEAPMEAN

#include <iostream>
#include <vector>

using namespace std;

int heapmean(vector<int> &heap){

    int total;
    total = 0;

    for(auto i = heap.begin() ; i != heap.end(); ++i){

        total += *i;
    }

    total = total/100;

    return total;
}

#endif
```

File: randgen.h

```
#ifndef RANDGEN
#define RANDGEN

#include <iostream>
#include <vector>

using namespace std;


void randGen(vector<int> &heap){

    int temp;

    srand (time(NULL));


    for(int c = 0; c < 100; c++){

        temp = rand() % (1000) + 1;
        heap.push_back(temp);

    }

}



#endif
```

One possible (due to randomization) terminal output is as follows:

```
./heap_example

Our Vector of 100 random integers
149 368 64 460 519 202 102 355 606 705 627 423 629 167 115 868 176 537
453 756 469 420 376 38 323 869 1000 406 643 475 198 143 843 613 602 713
815 704 67 772 760 46 194 388 212 660 607 740 196 59 495 17 831 222 54
153 442 405 910 84 231 107 226 73 720 828 137 886 883 204 9 642 601 554
29 164 213 987 903 761 397 749 777 227 970 830 379 412 234 289 495 464
395 73 889 466 900 377 703 782

The Original Heap

1000 987 910 903 970 900 643 886 815 830 889 782 869 442 475 868 883 713
704 777 705 495 660 740 495 831 629 406 405 231 226 720 843 613 602 642
554 460 606 772 760 469 519 412 420 627 607 466 703 323 64 17 202 222
54 153 149 102 167 84 115 107 198 73 143 828 137 355 176 204 9 368 601
537 29 164 213 67 453 761 397 749 756 227 46 194 379 388 234 289 212 464
395 73 376 423 38 377 196 59

We are adding a new integer of 462

Max of Heap is being deleted

Sorting the heap
987 970 910 903 889 900 643 886 815 830 660 782 869 442 475 868 883 713
704 777 705 495 627 740 495 831 629 406 405 231 226 720 843 613 602 642
554 460 606 772 760 469 519 412 420 464 607 466 703 462 64 17 202 222
54 153 149 102 167 84 115 107 198 73 143 828 137 355 176 204 9 368 601
537 29 164 213 67 453 761 397 749 756 227 46 194 379 388 234 289 212 323
395 73 376 423 38 377 196 59
```