ΟΝΟΜΑ: Μηλτιάδης
ΕΠΩΝΥΜΟ: Μαντές
ΑΜ: 1084661
ΕΤΟΣ: 2

10

ΑΣΚΗΣΗ ΕΡΓΑΣΤΗΡΙΟΥ : 3

- **ΣΧΟΛΙΑΣΜΟΣ ΛΥΣΗΣ**

Αρχικά, ορίζουμε μια συνάρτηση heapify_down() με ορίσματα ένα διάνυσμα <int> v και έναν ακέραιο i που εκφράζει τη θέση του πρώτου στοιχείου του σωρού μετά την εναλλαγή (20) και τύπο επιστροφής διάνυσμα <int>. Μέσα στο σώμα της συνάρτησης ορίζουμε τρία πεδία τύπου int που εκφράζουν το δεξί και αριστερό παιδί κάθε κόμβου καθώς και έναν δείκτη j. Αν το 2i είναι μικρότερο από το μέγεθος του heap μείον 1 (γιατί ξεκινάμε από τη θέση 1 του σωρού) τότε στο δεξί παιδί right κάθε κόμβου εκχωρούμε τη τιμή 2i + 1 και στο αριστερό παιδί left τη τιμή 2i. Αν το κλειδί του left είναι μικρότερο από το κλειδί του right τότε στο δείκτη j εκχωρούμε τη τιμή του left, ενώ αν είναι μεγαλύτερο τη τιμή του right. Αν το 2i είναι ίσο με το μέγεθος του heap μείον 1 τότε στο δείκτη j εκχωρούμε τη τιμή 2i. Αν το 2i είναι μεγαλύτερο από το μέγεθος του heap μείον 1 τότε αφήνουμε τον σωρό αμετάβλητο. Τέλος, αν η τιμή του σωρού στη θέση j είναι μικρότερη από τη θέση i τότε μέσω της swap() εναλλάσσουμε αυτά τα δύο στοιχεία και καλούμε ξανά αναδρομικά την heapify_down() με ορίσματα τους δείκτες i, j μέχρι το στοιχείο 20 να κατέλθει στη σωστή θέση. Μέσα στο σώμα της main() διαβάζουμε τις τιμές από το αρχείο και τις αποθηκεύουμε σε ένα διάνυσμα heap που ορίσαμε. Στη πρώτη θέση του heap εκχωρούμε μια τυχαία τιμή (0) μέσω της insert() προκειμένου να ξεκινήσουμε από τη θέση 1 του σωρού τη διαδικασία heapify down μετά την εναλλαγή των στοιχείων και τη διαγραφή του πρώτου στοιχείου που βρίσκεται στη τελευταία θέση. Για να γίνει αυτό αρχικά βρίσκουμε το ελάχιστο στοιχείο (2) και έπειτα με την iter_swap() εναλλάσσουμε το ελάχιστο στοιχείο (2) στη θέση 1 με το μέγιστο στοιχείο (20). Μετά την εναλλαγή διαγράφουμε με τη συνάρτηση pop_back() το στοιχείο 2 από τη τελευταία θέση και έτσι ο σωρός έχει μέγεθος 13. Εκχωρούμε το νέο σωρό σε ένα διάνυσμα <int> με όνομα h και καλούμε τη heapify_down() με ορίσματα h και 1, που είναι η θέση του στοιχείου 20, προκειμένου αυτό να κατέβει πιο κάτω και να έχουμε ξανά δομή σωρού στον h. Τέλος, για να βρούμε τη θέση του στοιχείου 20 μετά τη λήξη της διαδικασίας heapify down εκχωρούμε σε ένα int πεδίο toSearch τη τιμή 20 και με έναν βρόχο for διατρέχουμε το νέο σωρό και εκτυπώνουμε τη θέση στην οποία βρέθηκε αυτό.

- **ΚΩΔΙΚΑΣ ΛΥΣΗΣ**

```
#include <iostream>
#include <fstream>
#include <vector>
#include <conio.h>
#include <algorithm>

using namespace std;

void heapify_down(vector <int> &v, int i)
{
    int left;
    int right;
    int j;

    if(2*i>v.size() - 1)
    {
        return;
    }
    else if(2*i<v.size() - 1)
    {
        left=2*i;
        right=2*i+1;
        if(v[left]<v[right])
        {
            j=left;
        }
        else
        {
            j=right;
        }
    }
    else if(2*i==v.size() - 1)
    {
        j=2*i;
    }

    if(v[j]<v[i])
    {
        swap(v[j],v[i]);
        heapify_down(v,j);
    }
}

int main()
{
    vector <int> heap;
    int i;
    ifstream inputFile("Heap.txt");        // Input file
stream object
```

```
    if (inputFile.good())    // Check if file exists and
then open it.
     {
        heap.insert(heap.begin(), 0);
        int current_number = 0;
        while (inputFile >> current_number)    //Reads
data from file
        heap.push_back(current_number); //Stores them in
a vector

        inputFile.close();  //Close the file.

        //Display the numbers read
        i=heap.size();

        cout << "The numbers are: ";
        for (int count = 0; count < i; count++){
            cout << heap[count] << " ";
        }
        cout << endl;

    }else {
        cout << "Error in reading file!";
        exit(0);
    }

    // Finds the minimum element of the heap and prints
it
    cout << "the minimum element is: " <<
*min_element(heap.begin()+1,  heap.end());
    cout<<endl;

    //Move the last element into first place and first
element into last place
    iter_swap(heap.begin()+1, heap.end()-1);
    for (int j = 1; j < heap.size(); j++)
    {
        cout << heap[j] << " ";
    }
    cout<<endl;

    //Erases the last element from the heap
    heap.pop_back();
    for (int j = 1; j < heap.size(); j++)
    {
        cout << heap[j] << " ";
    }
    cout<<endl;

    vector <int> h=heap;
```

```
    int toSearch = heap[1];

    heapify_down(h, 1);
    cout<<"Min Heap\n";
    for (i = 1; i <= h.size()-1; i++)
    {
        cout<<h[i]<<endl;
    }

    for (i = 1; i <= h.size()-1; i++)
    {
        if (h[i] == toSearch)
        {
            cout << "Element " << toSearch << " is found
in position : " << i << endl;
        }
    }

    return 0;
}
```

- **ΑΠΟΤΕΛΕΣΜΑΤΑ ΕΚΤΕΛΕΣΗΣ**

```
The numbers are: 0 2 6 3 7 11 7 9 10 12 15 18 8 13 20
the minimum element is: 2
20 6 3 7 11 7 9 10 12 15 18 8 13 2
20 6 3 7 11 7 9 10 12 15 18 8 13
Min Heap
3
6
7
7
11
8
9
10
12
15
18
20
13
Element 20 is found in position : 12
```