МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №5

по дисциплине «Алгоритмы и структуры данных»

Тема: N-арная куча

Студентка гр. 9303 _____ Отмахова М.А.

Преподаватель _____ Филатов Ар.Ю.

Санкт-Петербург

2020

Цель работы.

Изучить структуру данных n-арная куча.

Основные теоретические положения.

Куча — это специализированная структура данных типа дерево, которая удовлетворяет свойству кучи: если B является узлом-потомком узла A, то ключ(A) ≥ ключ(B). Из этого следует, что элемент с наибольшим ключом всегда является корневым узлом кучи, поэтому иногда такие кучи называют max-кучами (в качестве альтернативы, если сравнение перевернуть, то наименьший элемент будет всегда корневым узлом, такие кучи называют min-кучами). Не существует никаких ограничений относительно того, сколько узлов-потомков имеет каждый узел кучи, хотя на практике их число обычно не более двух. Куча является максимально эффективной реализацией абстрактного типа данных, который называется очередью с приоритетом.

Формулировка задания.

Дан массив чисел и число n (n=1, 2, 3, …). Предполагая, что массив является n-арной кучей:

- Вывести его в виде n-арной кучи.

- Получить путь от корня до листа такой, что при каждом шаге вниз выбирается наибольший сын.

Выполнение работы.

В ходе выполнения лабораторной работы были написаны две функции `printHeat()` и `pathToLeaf()`.

Функция `printHeat()` принимает на вход массив, число n и размер массива. Функция выводит на консоль массив в виде n-арной кучи.

Функция `pathToLeaf()` принимает на вход индекс элемента, число n, массив чисел и размер данного массива. Данная функция является рекурсивной.

Внутри функции происходит поиск наибольшего сына для каждой вершины и вывод значения в консоль. После чего функция рекурсивно вызывает саму себя уже для данной вершины. Таким образом, получаем путь от корня до листа такой, что при каждом шаге вниз выбирается наибольший сын.

В функции `main()` происходит сначала считывание значения длины массива, далее через пробел вводится массив (подразумевается, что массив является n-арной кучей), после чего вводится значение n. А уже далее вызываются функции, описанные выше.

Выводы.

В ходе выполнения лабораторной работы была изучена структура данных куча. Была написана программа на языке C++, выводящая массив в идее кучи, а также выводит путь от корня до листа такой, что при каждом шаге вниз выбирается наибольший сын.

Исходный код разработанной программы представлен в приложении А.

Тестирование программы представлено в приложении Б.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

```
#include <iostream>

using namespace std;

void printHeat(int* num, int n, int size) {
    cout << "[ " << num[0] << " ]\n";
    int tmpInd = 1;
    int numOfBrackets = 1;
    while(tmpInd < size) {
        for(int j = 0; j < numOfBrackets; j++) {
            cout << '[';
            for (int i = tmpInd; i < tmpInd + n; i++)
{
                cout << ' ' << num[i] << ' ';
            }
            cout << ']';
            tmpInd += n;
        }
        cout << '\n';
        numOfBrackets *= n;
    }
    cout << '\n';
}

void pathToLeaf(int index, int n, int* num, int size)
{
    if (index*n+1 >= size) {
```

```
            return;
        }
        int max = num[index*n+1];
        int max_pos = index*n+1;
        for(int i = index*n + 1; (i <= index * n + n) &&
(i < size); i++ ) {
            if (num[i] > max) {
                max = num[i];
                max_pos = i;
            }
        }
        cout << max << ' ';
        pathToLeaf(max_pos, n, num, size);
    }

    int main() {
        int size;
        cout << "enter the number of array elements: " <<
endl;
        cin >> size;

        if(size<=0) {
            cout << "invalid data" << endl;
            return 0;
        }

        cout << "enter an array : " << endl;

        int* num = new int[size];
        for( int i = 0; i < size; i++) {
```

```
            cin >> num[i];
        }

        int n;
        cout << "enter value n: " << endl;
        cin >> n;

        if(n<=0) {
            cout << "invalid data" << endl;
            return 0;
        }

        cout << "heap looks something like this: " <<
endl ;
        printHeat(num, n, size);

        cout << "path to leaf: ";

        cout << num[0] << ' ';
        pathToLeaf(0, n, num, size);

        delete[] num;
        return 0;
    }
```

ПРИЛОЖЕНИЕ Б

ТЕСТИРОВАНИЕ ПРОГРАММЫ

Тестирование программы представлено в таблице 1.

Таблица 1 - Тестирование программы

Входные данные	Результат работы программы
enter the number of array elements: 7 enter an array : 15 14 13 12 11 3 2 enter value n: 2	heap looks something like this: [15] [14 13] [12 11][3 2] path to leaf: 15 14 12
enter the number of array elements: 13 enter an array : 100 99 98 97 96 95 94 93 92 91 90 89 enter value n: 3	heap looks something like this: [100] [99 98 97] [96 95 94][93 92 91][90 89 88] path to leaf: 100 99 96
enter the number of array elements: 13 enter an array : 15 12 13 14 10 7 6 1 2 3 9 8 5 enter value n: 3	heap looks something like this: [15] [12 13 14] [10 7 6][1 2 3][9 8 5] path to leaf: 15 14 9
enter the number of array elements: 21 enter an array : 17 14 12 13 11 1 2 3 4 6 3 4 5 9 7 6 enter value n: 4	heap looks something like this: [17] [14 12 13 11] [1 2 3 4][6 3 4 5][9 7 6 5][2 path to leaf: 17 14 4