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

ОТЧЕТ

по лабораторной работе №1
по дисциплине «Алгоритмы и структуры данных»
Тема: Рекурсия

Студент гр. 8304 _____ Алтухов А.Д.

Преподаватель _____ Фиалковский М.С.

Санкт-Петербург
2019

Цель работы.

Освоение рекурсивного метода решения задач.

Задание.

Вариант 1.

Для заданных неотрицательных целых n и m вычислить (рекурсивно) биномиальные коэффициенты, пользуясь их определением:

$$C_n^m = \begin{cases} 1, \text{если } m = 0, n > 0 \text{ или } m = n \geq 0, \\ 0, \text{если } m > n \geq 0, \\ C_{n-1}^{m-1} + C_{n-1}^m \text{ в остальных случаях.} \end{cases}$$

Описание алгоритма работы.

Программа использует следующий алгоритм для выполнения задачи: функция, обрабатывающая сочетание, получает на вход необходимые данные и ссылку на переменную, предназначенную для хранения результата. Далее, пользуясь определением сочетания, к переменной отвечающей за результат прибавляется или прибавляется один, или не прибавляется ничего, или происходит два рекурсивных вызова функции с измененными следовательно определению параметрами. Переменная, хранящая результат, необходима для того, чтобы уменьшить время выполнения программы: результат начнет считаться уже во время рекурсивных вызовов, а не по их завершению.

Описание функций и структур.

1. Написана функция int main(int argc,char* argv[]) со стандартной сигнатурой, отвечающая за открытие необходимых для ввода-вывода файлов, общение с пользователем и запуск функции-обработчика.

2. Основная функция void recBinomial(int n, int m, unsigned long long int& res, int depth, ofstream& f) ничего не возвращает, так как результат вычисления хранится в передаваемой функции переменной, принимает числа n и m — сочетание из n по m, ссылку на переменную res, сохраняющую результат вычислений, переменную depth, отображающую глубину рекурсии, ссылку на

объект для операций с файлом для вывода. Эта функция либо прибавляет к res единицу, либо ничего не делает, либо вызывает себя рекурсивно с параметрами n-1, m-1, res, depth + 1, f и n-1, m, res, depth + 1, f следовательно определению сочетания.

3. Функция void inputProcessing(int n, int m, ofstream& f) получает считанные данные, проверяет их на корректность и, если все в порядке, запускает функцию recBinomial для подсчета результата.

4. Функция string createTabs(int depth) возвращает строку из depth числа символов табуляции.

3. Вспомогательные функции void fRead(ifstream& f, int& n, int& m) и void consoleInput(int& n, int& m) ничего не возвращают и служат для ввода входных данных с файла и консоли соответственно. Принимают ссылки на переменные, созданные в main’е, необходимые для хранения входных данных.

Тестирование.

Некорректные данные:

№	n	m	Ответ программы	Ожидаемый ответ
1	-1	0	Введенные данные некорректны	Сообщение об ошибке
2	50	-7	Введенные данные некорректны	Сообщение об ошибке

Корректные данные:

№	n	m	Ответ программы	Ожидаемый ответ
3	1	0	1	1
4	100	100	1	1
5	5	3	10	10

6	10	5	252	252
7	0	1	0	0

Выводы.

В ходе выполнения лабораторной работы была написана рекурсивная функция вычисления числа сочетаний с использованием определения, как было предложено в задании.

ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД.

```
#include <iostream>
#include <fstream>
#include <climits> //для отслеживания переполнения
#include <string>
using namespace std;

bool fRead(ifstream& f, int& n, int& m) { //вспомогательная функция для считывания входных данных с файла

	bool res = false;
	if ((f >> n) && (f >> m))
		res = true;
	char buff[50];
	f.getline(buff, 50); //досчитываем строку
	return res;

}

void consoleInput(int& n, int& m) { //вспомогательная функция для считывания входных данных с консоли
	cout << "Введите n и m.\n";
	cin >> n >> m;
}

string createTabs(int depth){ //возвращает строку с нужным колвом табов
	string res = "";
	for (int i=0; i<depth; i++)
		res+="\t";
```

```
    return res;
}

void recBinomial(int n, int m, unsigned long long int& res, int depth, ofstream& f) { //считаем биномиалы

    cout << createTabs(depth) << "Начало. Глубина: " << depth << ", " << "n: " << n << ", m: " << m << ", Результат: " << res << "\n";
    f << createTabs(depth) << "Начало. Глубина: " << depth << ", " << "n: " << n << ", m: " << m << ", Результат: " << res << "\n";
    if (ULLONG_MAX == res) { //переполнение
        cout << createTabs(depth) << "Достигнуто максимальное значение. Конечный результат может быть не верен. Глубина: " << depth << ", " << "n: " << n << ", m: " << m << ", Результат: " << res << "\n";
        f << createTabs(depth) << "Достигнуто максимальное значение. Конечный результат может быть не верен. Глубина: " << depth << ", " << "n: " << n << ", m: " << m << ", Результат: " << res << "\n";
        return;
    }
    if (((m == 0) && (n > 0)) || ((m == n) && (m >= 0))) {

        res += 1;
        cout << createTabs(depth) << "+1. Глубина: " << depth << ", " << "n: " << n << ", m: " << m << ", Результат: " << res << "\n";
        f << createTabs(depth) << "+1. Глубина: " << depth << ", " << "n: " << n << ", m: " << m << ", Результат: " << res << "\n";
        return;
    }
    if ((m > n) && (m >= 0)) {
```

```
                cout << createTabs(depth) << "+0. Глубина: " << depth << ", "
<< "n: " << n << ", m: " << m << ", Результат: " << res << "\n";
                f << createTabs(depth) << "+0. Глубина: " << depth << ", " <<
"n: " << n << ", m: " << m << ", Результат: " << res << "\n";
                return;
            }
            //если не случаи выше, то по определению разложение на сумму
двух сочетаний
            recBinomial(n - 1, m - 1, res, depth + 1, f);
            recBinomial(n - 1, m, res, depth + 1, f);
            cout << createTabs(depth) << "Конец. Глубина: " << depth << ", " <<
"n: " << n << ", m: " << m << ", Результат: " << res << "\n";
        }

        void inputProcessing(int n, int m, ofstream& f){

            if (!((n >= 0) && (m >= 0))) {
                cout << "Введенные данные некорректны: " << n << " " << m
<< "\n\n\n";
                f << "Введенные данные некорректны: " << n << " " << m <<
"\n\n\n";
                return;
            }
            cout << "Введенные данные: " << n << " " << m << "\n";
            f << "Введенные данные:" << n << " " << m << "\n";

            unsigned long long int res = 0; //кол-во сочетаний >=0
            recBinomial(n, m, res, 0, f);
            cout << "Результат: " << res << "\n\n\n";
            f << "Результат: " << res << "\n\n\n";
```

```
}

int main(int argc,char* argv[]) {

	ifstream inputF;
	if (argc > 1)
		inputF.open(argv[1]);
	else
		inputF.open("input.txt");



	//ifstream inputF("input.txt");
	ofstream outputF("output.txt");
	if (!(outputF.is_open())) {
		cout << "Файл вывода не найден.\n";
		return 0;
	}
	if (!(inputF.is_open()))
		cout << "Файл ввода не найден.\n";

	int n = 0, m = 0;
	int inputType = 0;
	if (argc == 1){
		cout << "Выберите тип ввода:\n1 - ввод из консоли.\n2 - ввод из файла.\n";
		cin >> inputType;
	}
	else
		inputType = 2;
```

```
if (inputType == 1) //выбран ввод с консоли
	consoleInput(n, m);
else if (inputType == 2) { //выбран ввод с файла
	if (!(inputF.is_open())) {
		cout << "Файл ввода не найден.\n";
		return 0;
	}
}
else {
	cout << "Некорректный тип ввода. Выбран тип по умолчанию.\n"; //ввод с консоли
	consoleInput(n, m);
}

if (inputType == 1){
	inputProcessing(n, m, outputF);

}
else{
	while (fRead(inputF, n, m)){ //читаем пока в файле что-то есть
		inputProcessing(n, m, outputF);
	}
}


inputF.close();
outputF.close();
```

```
    return 0;
}
```