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

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Алгоритмы и структуры данных»

Тема: Программирование рекурсивных алгоритмов

Студент гр. 9303 _____ Лойконен М.Р.

Преподаватель _____ Филатов А.Ю.

Санкт-Петербург
2020

Цель работы.

Ознакомиться с основными понятиями и приемами рекурсивного программирования. Научиться решать задачи при помощи рекурсивных функций.

Задание.

Вариант 11.

Написать программу, которая по заданному константному_выражению вычисляет его значение либо сообщает о переполнении (превышении заданного значения) в процессе вычислений.

константное_выражение::= ряд_цифр|

константное_выражение знак_операции константное_выражение

знак_операции::= +| – | *

ряд_цифр::= цифра | цифра ряд_цифр

Основные теоретические положения.

Рекурсивным называется объект, содержащий сам себя или определенный с помощью самого себя.

Мощность рекурсии связана с тем, что она позволяет определить бесконечное множество объектов с помощью конечного высказывания. Точно так же бесконечные вычисления можно описать с помощью конечной рекурсивной программы. Рекурсивные алгоритмы лучше всего использовать, когда решаемая задача, вычисляемая функция или обрабатываемая структура данных определены с помощью рекурсии.

Если процедура (функция) P содержит явное обращение к самой себе, она называется прямо рекурсивной. Если P содержит обращение к процедуре (функции) Q, которая содержит (прямо или косвенно) обращение к P, то P называется косвенно рекурсивной.

Выполнение работы.

`int calculate(string s, int ind, int depth, int& err)` — прямо рекурсивная функция. Принимает на вход строку `s`, содержащую исходное константное выражение, число `ind`, являющееся индексом, с которого надо начать считывать следующий элемент строки, число `depth` — глубина рекурсии и ссылку на `err` — переменная для обозначения ошибки (1 — произошла ошибка, 0 — ошибок не было). Функция рекурсивно считает значение по константному выражению, если во время вычислений происходит переполнение, то в переменную `err` записывается значение 1, и функция завершает свою работу. Функция возвращает целое число, в которое записан результат вычислений.

`int strToDigit(string s, int start, int end, int& err)` — функция считывает число из переданной строки `s`. Переменные `start` и `end` — индексы, соответственно начальный и конечный, указывающие в строке `s` на промежуток, содержащий число, которое необходимо считать. Ссылка на `err` — переменная для обозначения ошибки. Число записывается в строку `str` и преобразуется к числу при помощи функции `stoi`. При использовании `stoi`, функция отлавливает исключения `out_of_range` или `invalid_argument`. Функция возвращает целое число.

`void printSpace(int depth)` — функция печатает « » для обозначения уровня рекурсии при выводе промежуточных данных.

Все данные программа считывает из файла, результат вычислений или сообщение об ошибке также записывается в файл. Все промежуточные данные выводятся в консоль.

Разработанный программный код см. в приложении А.

Результаты тестирования см. в приложении Б.

Выводы.

Было изучено понятие рекурсия.

Была написана программа, которая вычисляет значение заданного константного выражения или сообщает о переполнении во время вычислений, используя прямо рекурсивную функцию. Константное выражение программа считывает из файла, промежуточные данные, в том числе глубина рекурсии, выводятся в консоль, а результат вычислений и в консоль, и в файл.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

```
#include <iostream>
#include <fstream>
#include <string>
#include <climits>

using namespace std;

int strToDigit(string s, int start, int end, int& err){
	int size = end - start + 1;
	int d = 0;
	string str;
	for (int i = 0; i<size; i++){
		str[i] = s[start];
		start++;
	}
	str[size] = '\0';

	try{
		d = stoi(str);
	}
	catch(out_of_range){
		cout << "Одно из введенных чисел превышает максимально допустимое значение типа int\n";
		err = 1;
		return 0;
	}
	catch(invalid_argument){
		cout << "Строка не соответсвует константному выражению\n";
		err = 1;
		return 0;
	}
	return d;
}

void printSpace(int depth){
	for (int i = 0; i<depth; i++){
		cout << " ";
	}
}


int calculate(string s, int ind, int depth, int& err){
	printSpace(depth);
	cout << "Вызов calculate(" << depth << "): Глубина рекурсии: " << depth << ".\n";
	int res = 0, d1 = 0, d2 = 0, countSym = 0, end = 0;

	for (int i = ind; i<s.length(); i++){
		if (s[i] == '+'){
```

```
                        end = i-1;
                        d1 = strToDigit(s, end - countSym + 1, end, err);

                        if (err){
                              return 0;
                        }

                        countSym = 0;
                        d2 = calculate(s, i+1, depth+1, err);

                        if ((d2 > 0 && d1 > INT_MAX - d2) || (d2 < 0 &&
d1 < INT_MIN - d2)){
                              cout  <<  "Во  время  вычислений  произошло
переполнение типа int\n";
                              err = 1;
                        }
                        else{
                              res = d1 + d2;
                        }

                        if (err){
                              return 0;
                        }

                        printSpace(depth);
                        cout << "Завершение calculate(" << depth << ").
Получившееся выражение: " << d1 << " + " << d2 << " = " << res << '\
n';
                        return res;
                  }
                  if (s[i] == '-'){
                        end = i-1;
                        d1 = strToDigit(s, end - countSym + 1, end, err);

                        if (err){
                              return 0;
                        }

                        countSym = 0;
                        d2 = calculate(s, i+1, depth+1, err);
                        if ((d2 > 0 && d1 < INT_MIN + d2) || (d2 < 0 &&
d1 > INT_MAX + d2)){
                              cout  <<  "Во  время  вычислений  произошло
переполнение типа int\n";
                              err = 1;
                        }
                        else{
                              res = d1 - d2;
                        }

                        if (err){
                              return 0;
                        }

                        printSpace(depth);
```

```
                    cout << "Завершение calculate(" << depth << "). Получившееся выражение: " << d1 << " - " << d2 << " = " << res << '\n';
                    return res;
                }
                if (s[i] == '*'){
                    end = i-1;
                    d1 = strToDigit(s, end - countSym + 1, end, err);

                    if (err){
                        return 0;
                    }

                    countSym = 0;
                    d2 = calculate(s, i+1, depth+1, err);

                    if (d1 != 0){
                        if (d2 > 0){
                            if (d1 > INT_MAX/d2){
                                cout << "Во время вычислений произошло переполнение типа int\n";
                                err = 1;
                            }
                        }
                        else{
                            if (d2 < INT_MIN/d1){
                                cout << "Во время вычислений произошло переполнение типа int\n";
                                err = 1;
                            }
                        }
                    }

                    if (err){
                        return 0;
                    }

                    res = d1*d2;
                    printSpace(depth);
                    cout << "Завершение calculate(" << depth << "). Получившееся выражение: " << d1 << " * " << d2 << " = " << res << '\n';
                    return res;
                }
                countSym++;
            }

            res = strToDigit(s, s.length()-countSym, s.length()-1, err);

            if (err){
                return 0;
            }

            printSpace(depth);
            cout << "Завершение calculate(" << depth << "). Получившееся выражение: " << res << '\n';
```

```
        return res;
    }

    int main(){
        int depth = 0, res = 0, err = 0;
        string s;
        ifstream fin("tests.txt");
        ofstream fout("results.txt");
        while (fin >> s){
            cout << "Константное выражение: " << s << '\n';
            res = calculate(s, 0, depth, err);
            if (err){
                cout << "При выполнении работы программы
произошла ошибка\n\n";
                fout << "При выполнении работы программы
произошла ошибка\n";
                err = 0;
                continue;
            }
            cout << "Результат вычислений: " << res << "\n\n";
            fout << "Результат вычислений: " << res << '\n';
        }
        return 0;
    }
```

ПРИЛОЖЕНИЕ Б
ТЕСТИРОВАНИЕ

Таблица Б.1 - Примеры тестовых случаев

№ п/п	Входные данные	Выходные данные	Комментарии
1.	1+2*10-11	Вызов calculate(0): Глубина рекурсии: 0. Вызов calculate(1): Глубина рекурсии: 1. Вызов calculate(2): Глубина рекурсии: 2. Вызов calculate(3): Глубина рекурсии: 3. Завершение calculate(3). Получившееся выражение: 11 Завершение calculate(2). Получившееся выражение: 10 - 11 = -1 Завершение calculate(1). Получившееся выражение: 2 * -1 = -2 Завершение calculate(0). Получившееся выражение: 1 + -2 = -1 Результат	Корректное выражение

		вычислений: -1	
2.	20*214748364+1-678	Вызов calculate(0): Глубина рекурсии: 0. Вызов calculate(1): Глубина рекурсии: 1. Вызов calculate(2): Глубина рекурсии: 2. Вызов calculate(3): Глубина рекурсии: 3. Завершение calculate(3). Получившееся выражение: 678 Завершение calculate(2). Получившееся выражение: 1 - 678 = -677 Завершение calculate(1). Получившееся выражение: 214748364 + -677 = 214747687 Во время вычислений произошло переполнение типа int При выполнении работы программы произошла ошибка	Переполнение при вычислении.

3.	+af32f	Вызов calculate(0): Глубина рекурсии: 0. Строка не соответствует константному выражению При выполнении работы программы произошла ошибка	Некорректное выражение.
4.	990233+877-1100	Вызов calculate(0): Глубина рекурсии: 0. Вызов calculate(1): Глубина рекурсии: 1. Вызов calculate(2): Глубина рекурсии: 2. Завершение calculate(2). Получившееся выражение: 1100 Завершение calculate(1). Получившееся выражение: 877 - 1100 = -223 Завершение calculate(0). Получившееся выражение: 990233 + -223 = 990010 Результат вычислений: 990010	Корректное выражение.
5.	8-12425*10-777*0	Вызов calculate(0): Глубина рекурсии: 0.	Корректное выражение.

		Вызов calculate(1): Глубина рекурсии: 1. Вызов calculate(2): Глубина рекурсии: 2. Вызов calculate(3): Глубина рекурсии: 3. Вызов calculate(4): Глубина рекурсии: 4. Завершение calculate(4). Получившееся выражение: 0 Завершение calculate(3). Получившееся выражение: 777 * 0 = 0 Завершение calculate(2). Получившееся выражение: 10 - 0 = 10 Завершение calculate(1). Получившееся выражение: 12425 * 10 = 124250 Завершение calculate(0). Получившееся выражение: 8 -	

		124250 = -124242 Результат вычислений: -124242	