**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №3**

**по дисциплине «Алгоритмы и структуры данных»**

Тема: Стеки и очереди

|  |  |  |
| --- | --- | --- |
| Студентка гр. 7382 |  | Головина Е.С. |
| Преподаватель |  | Фирсов М.А. |

Санкт-Петербург

2018

**Цель работы.**

Изучить работу со стеком. Научиться применять стек для выполнения требуемых операций.

**Задание.**

Вариант 7-д:

В заданном текстовом файле *F* записана формула вида

< формула > ::= < цифра > | *М*  ( < формула > ,  < формула > ) |

*m*  ( < формула > ,  < формула > )

< цифра > ::= 0 | 1 | ... | 9

где *M* обозначает функцию *max*, а *m* – функцию *min*. Вычислить (как целое число) значение данной формулы. Например, *M* (5, *m*(6, 8)) = 6.

**Постановка задачи.**

Нужно написать программу, которая считывает формулу, записывает все операции и цифры в стек, и затем происходит подсчет значения или вывод сообщения об ошибке, если формула была введена некорректно.

**Описание алгоритма.**

1. Программа запрашивает как пользователь хочет ввести данные: из файла или из терминала. Если из файла — п.2, если из терминала — п.3.

2. Программа запрашивает название файла, данные откуда надо взять и использует его в дальнейшем, если он существует. Далее п.4.

3. Для однородности программы в дальнейшем программа считывает строку и записывает ее в файл, который дальше будет использовать. Далее п.4.

4. Начинается запись букв и цифр в стек, пока не будет достигнут конец файла.

5. Происходит вывод содержимого стека после его заполнения для наглядности.

6. Программа подсчитывает количество элементов в стеке и начинает подсчет результата, заходим в цикл.

7. Проверяем содержимое верхнего элемента стека, если оно равно 'm' — п.8, если оно равно 'M' — п.9, если другое — п.10. Проверка происходит пока не достигнут конец стека, в этом случае цикл завершается (п.11).

8. Если min связана с 2мя цифрами, то программа посчитает значение и запишет его в соответствующую ячейку массива num, использующегося для промежуточных результатов и переходит в п.7, иначе напишет сообщение об ошибке и программа прервется.

9. Если max связана с 2мя цифрами, то программа посчитает значение и запишет его в соответствующую ячейку массива num, использующегося для промежуточных результатов и переходит в п.7, иначе напишет сообщение об ошибке и программа прервется.

10. Проверяется, является ли значение из стека цифрой, если да, программа записывает его в соответствующую ячейку массива num, использующегося для промежуточных результатов, иначе напишет сообщение об ошибке и программа прервется.

11. Программа выводит результат вычислений при корректном подсчете значения.

**Методы класса и структуры данных:**

1. struct StackElem{

Type value;

StackElem \* next;

};

value — значение хранящееся в стеке;

next — указатель на следующий элемент.

2. **void Stack::push(Type val);**

val — значение добавляемое в стек.

Метод кладет в стек элемент и ничего не возвращает.

3. **Type Stack::pop();**

Метод вытаскивает из стека элемент и очищает память, выделенную под него. Возвращает значение, хранящееся в элементе стека.

**4. bool Stack::isEmpty();**

Метод проверяет стек на пустоту. Если он пуст возвращает true, иначе false.

**5. int Stack::count();**

Метод считает количество элементов в стеке и возвращает их количество.

**6. void Stack::print();**

Метод печатает значения всех элементов в стеке в порядке обращения к ним, если стек не пустой. Ничего не возвращает.

Тестирование.

Было сделано 10 тестов для демонстрации и проверки работы программы.

|  |  |  |
| --- | --- | --- |
| Тест № | Данные | Результат |
| 1 | M(m(M(9,7),M(1,5)),M(m(2,3),m(1,7))) | Result = 5 |
| 2 | M(9,m(1,7)) | Result = 9 |
| 3 | M(M(1,1),m(1,1)) | Result = 1 |
| 4 | M(5,m(2,7)) | Result = 5 |
| 5 | 5 | Result = 5 |
| 6 | ghghghghg | Incorrect data!(digit) |
| 7 | M(m(M())) | Incorrect data!(M) |
| 8 | M(7,m(4)) | Incorrect data!(m) |
| 9 | M(M(5),8) | Incorrect data!(M) |
| 10 | MM(5,8) | Incorrect data!(M) |

Для более наглядной демонстрации работы программы был создан bash-скрипт, последовательно выводящий содержимое очередного теста и результат работы программы для этого теста. Код bash-скрипта представлен в приложении Б, результат работы скрипта — для тестов 1 и 9 представлен в приложении В.

Выберем 2 тест для детального обзора. Программа всегда использует текстовый файл для обработки строки и записывания элементов в стек.

1. Программа идет с начала строки и если символ буква или цифра записывает его в стек. Надо добавить, что программа на этапе записи в стек не проверяет корректность введенной формулы. Для строки **M(9,m(1,7))** :

Проверяем является ли «М» буквой или цифрой. Записываем в стек.

Проверяем является ли «(» буквой или цифрой. Не записываем в стек.

Проверяем является ли «9» буквой или цифрой. Записываем в стек.

Проверяем является ли «,» буквой или цифрой. Не записываем в стек.

Проверяем является ли «m» буквой или цифрой. Записываем в стек.

Проверяем является ли «(» буквой или цифрой. Не записываем в стек.

Проверяем является ли «1» буквой или цифрой. Записываем в стек.

Проверяем является ли «,» буквой или цифрой. Не записываем в стек.

Проверяем является ли «7» буквой или цифрой. Записываем в стек.

Проверяем является ли «)» буквой или цифрой. Не записываем в стек.

Проверяем является ли «)» буквой или цифрой. Не записываем в стек.

Больше символов нет. Запись в стек завершена. Полученный стек:

7

1

m

9

M

2. После того как все записано. Выполняется вычисление значения, пока не закончатся элементы в строке. Числа для вычислений сохраняем в массиве int num[3], пока не встретится операция min(m) или max(M).

Элемент 7. Запись в num[0]. Убираем его из стека.

Элемент 1. Запись в num[1]. Убираем его из стека.

Элемент m. Вычисление min(1,7). Запись значения(1) в num[0]. Убираем «m» из стека.

Элемент 9. Запись в num[1]. Убираем его из стека.

Элемент M. Вычисление Max(9,1). Запись значения(9) в num[0]. Убираем «M» из стека.

3. Так как не было никаких ошибок в синтаксисе формулы и посчитать ее значение оказалось возможным, выводим результат вычисления = 9.

**Вывод**

В ходе лабораторной работы были изучены принципы работы со стеком, его заполнение, извлечение элементов из него и использование его для выполнения операций.

**ПРИЛОЖЕНИЕ А**

**ИСХОДНЫЙ КОД, СОДЕРЖАЩИЙ ГОЛОВНУЮ ФУНКЦИЮ**

#include <iostream>

#include <cstdio>

#include <cctype>

#include <cstring>

#define N 500

template <class Type>

struct Stack {

//constructor

Stack():topElem(0)

{}

//destructor

~Stack()

{

StackElem \* temp=topElem;

while(temp != NULL){

topElem=topElem->next;

delete temp;

temp=topElem;

}

}

//add element

void push(Type val){

StackElem\* temp;

temp = new StackElem;

temp->value=val;

temp->next=topElem;

topElem=temp;

}

//remove element

Type pop(){

StackElem \* temp=topElem;

if (topElem){

Type val = topElem->value;

topElem = topElem->next;

delete temp;

return val;

}

else{

std::cout << "Stack is empty!" << std::endl;

exit(1);

}

}

//check emptiness

bool isEmpty(){

if (topElem==NULL) return true;

else return false;

}

//take top element

Type top(){

return topElem->value;

}

//count quantity of elements in stack

int count(){

StackElem \* temp=topElem;

int counter=0;

while (temp!=NULL){

counter++;

temp=temp->next;

}

return counter;

}

//print elements

void print(){

StackElem \* temp=topElem;

while(temp!=NULL){

std::cout << "\t" << temp->value << std::endl;

temp=temp->next;

}

}

struct StackElem{

Type value;

StackElem \* next;

};

StackElem \* topElem;

};

int main(){

char symbol;

int \* num = new int[3];

num[0]=-1;num[1]=-1;num[2]=-1;

char\* str = new char[N];

int index=0,counter=0;

FILE \* txt\_file;

//choice of terminal or file mode

std::cout << "Type 1 to proceed in terminal, and 2 to use file.\n(else you will be forced to use file)" << std::endl;

std::cin.getline(str,2);

symbol=str[0];

if (symbol=='1'){

std::cin.getline(str,N);

txt\_file = fopen("../in.txt","w+");

fputs(str,txt\_file);

fclose(txt\_file);

txt\_file = fopen("../in.txt","r");

}

else{

std::cout << "Please, type the name of the file in format <name>.txt" << std::endl;

std::cin.getline(str,N);

str[strlen(str)]='\0';

txt\_file = fopen(str,"r");

if (txt\_file == NULL){

printf("File not existed!\n");

return 0;

}

}

//output of the string

fpos\_t txt\_pointer;

fgetpos(txt\_file,&txt\_pointer);

std::cout << "RECIEVED FORMULA:" << std::endl;

fgets(str,N,txt\_file);

std::cout << str << std::endl;

fsetpos(txt\_file,&txt\_pointer);

//creating stack with values of type char

Stack<char> new\_stack;

//filling stack with digits and operations(m,M)

while(1){

if (feof(txt\_file)) break;

symbol=fgetc(txt\_file);

if (isdigit(symbol) || isalpha(symbol)){

new\_stack.push(symbol);

}

}

fclose(txt\_file);

//output of stack

std::cout << "Stack:" <<std::endl;

new\_stack.print();

std::cout << "Steps:" <<std::endl;

counter=new\_stack.count();

//counting of the result

while(counter){

switch (new\_stack.top()){

case 'm':

if (index-2<0){

std::cout << "Incorrect data!(m)" << std::endl;

return 0;

}

std::cout<<"Min("<<num[index-1]<<","<<num[index-2]<<") = ";

new\_stack.pop();

num[index-2]=std::min(num[index-1],num[index-2]);

std::cout << num[index-2] << std::endl;

index--;

break;

case 'M':

if (index-2<0){

std::cout << "Incorrect data!(M)" << std::endl;

return 0;

}

std::cout<<"Max("<<num[index-1]<<","<<num[index-2]<<") = ";

new\_stack.pop();

num[index-2]=std::max(num[index-1],num[index-2]);

std::cout << num[index-2] << std::endl;

index--;

break;

default:

if (isdigit(new\_stack.top())){

num[index]=new\_stack.pop() - '0';

index++;

}

else{

std::cout << "Incorrect data!(digit)" << std::endl;

return 0;

}

break;

}

counter--;

}

std::cout << "Result = " << num[0] << std::endl;

remove("in.txt");

delete [] str;

delete [] num;

return 0;

}

**ПРИЛОЖЕНИЕ Б**

**ИСХОДНЫЙ КОД СКРИПТА**

#!/bin/bash

g++ Source/lab3.cpp

echo " "

echo "Tests with correct data:"

echo " "

echo "Test 1"

echo " "

./a.out <./Tests/test1.txt

echo " "

echo "Test 2"

echo " "

./a.out <./Tests/test2.txt

echo " "

echo "Test 3"

echo " "

./a.out <./Tests/test3.txt

echo " "

echo "Test 4"

echo " "

./a.out <./Tests/test4.txt

echo " "

echo "Test 5"

echo " "

./a.out <./Tests/test5.txt

echo " "

echo "Tests with incorrect data:"

echo " "

echo " "

echo "Test 6"

echo " "

./a.out <./Tests/test6.txt

echo " "

echo "Test 7"

echo " "

./a.out <./Tests/test7.txt

echo " "

echo "Test 8"

echo " "

./a.out <./Tests/test8.txt

echo " "

echo "Test 9"

echo " "

./a.out <./Tests/test9.txt

echo " "

echo "Test 10"

echo " "

./a.out <./Tests/test10.txt

rm -f a.out

**ПРИЛОЖЕНИЕ В**

**РЕЗУЛЬТАТ РАБОТЫ СКРИПТА**

Test 2

RECIEVED FORMULA:

M(9,m(1,7))

Stack:

7

1

m

9

M

Steps:

Min(1,7) = 1

Max(9,1) = 9

Result = 9

Test 9

RECIEVED FORMULA:

M(M(5),8)

Stack:

8

5

M

M

Steps:

Max(5,8) = 8

Incorrect data!(M)