**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №2**

**по дисциплине «Программирование»**

Тема: **Динамические структуры данных.**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6304 |  | Виноградов К.А. |
| Преподаватель |  | Кринкин К.В. |

Санкт-Петербург

2017

Цель:

Написание программы - стековой машины.

Задание:

Напишите программу, На вход которой подается последовательность (не более 100 элементов) из чисел и арифметических операций (+, -, \*, / (деление нацело)) разделенных пробелом, которые программа должна интерпретировать и выполнить по следующим правилам:

Если очередной элемент входной последовательности - число, то положить его в стек

Если очередной элемент - знак операции, то применить эту операцию над двумя верхними элементами стека, а результат положить обратно в стек (следует считать, что левый операнд выражения лежит в стеке глубже)

Если входная последовательность закончилась, то вывести результат (число в стеке)

Если в процессе вычисления возникает ошибка (для операции в стеке не хватает аргументов или по завершении работы программы в стеке более одного элемента), то вместо результата следует вывести "error"

Стек требуется реализовать самостоятельно на базе массива.

Ход работы:

* Разбиение исходного текста на слова

char\* token=strtok(origin, " "); //

while(token!=NULL) //

{ //

int numb=strlen(token); // деление массива origin

strncpy(nums[i], token, numb); // на токены

token=strtok(NULL, " "); //

i++; //

} //

* Перевод символов в числа

for(j=0;j<=i;j++){ // цикл перевода цифр из

inums[j]=atoi(nums[j]); // типа char в int

} // (символы операций переводятся как 0)

* Выявление символов операций (в данном случае сложения)

if(strcmp(nums[k], plus)==0){ // (\*) обращаемся к токену

if(j>=2){ // проверка элементов

stack[j-2]=stack[j-2]+stack[j-1]; // операция с ними

stack[j-1]=0; // обнуление ненужного

j-=2; // откат счетчика

k++; //

} //

else{ // иначе

m=1; // переключение триггера

j=i; // завершение цикла

} //

} // (\*) ======================

Вывод:

В данной лабораторной работе мы познакомились с принципом работы стековой машины, а также написали программу выполняющую данный алгоритм.

ПРИЛОЖЕНИЕ 1

#include <stdio.h> //

#include <stdlib.h> // подключение стандартных библиотек

#include <string.h> //

int main() {

int i=0; //

int j=0; // счетчики

int k=0; //

int m=0; // триггер

char plus[2]="+"; //

char minus[2]="-"; // массивы с символами операций

char umn[2]="\*"; // (для использования в strcmp)

char del[2]="/"; //

char origin[100]; // массив для исходной последовательности

char nums[100][100]; // массив для токенов из origin

int inums[100]; // массив для токенов, преобразованных в int

int stack[100]; // стековый массив

fgets(origin, 100, stdin); // счет с потока ввода

char\* token=strtok(origin, " "); //

while(token!=NULL) //

{ //

int numb=strlen(token); // деление массива origin

strncpy(nums[i], token, numb); // на токены

token=strtok(NULL, " "); //

i++; //

} //

for(j=0;j<=i;j++){ // цикл перевода цифр из

inums[j]=atoi(nums[j]); // типа char в int

} // (символы операций переводятся как 0)

for(j=0;j<=i;j++){ //

if(inums[k]!=0){ // выявление символов операций

stack[j]=inums[k]; // (если число, то записывается в стек,

k++; // иначе – символ операции)

} //

else{

if(strcmp(nums[k], plus)==0){ // (\*) обращаемся к токену ===

if(j>=2){ // проверка элементов

stack[j-2]=stack[j-2]+stack[j-1]; // операция с ними

stack[j-1]=0; // обнуление ненужного

j-=2; // откат счетчика

k++; //

} //

else{ // иначе

m=1; // переключение триггера

j=i; // завершение цикла

} //

} // (\*) ======================

else if(strcmp(nums[k], minus)==0){ // ===================

if(j>=2){ // идентично (\*)

stack[j-2]=stack[j-2]-stack[j-1]; //

stack[j-1]=0; //

j-=2; //

k++; //

} //

else{ //

m=1; //

j=i; //

} //

} // =======================

else if(strcmp(nums[k], umn)==0){ // =======================

if(j>=2){ // идентично (\*)

stack[j-2]=stack[j-2]\*stack[j-1]; //

stack[j-1]=0; //

j-=2; //

k++; //

} //

else{ //

m=1; //

j=i; //

} // =====================

else if(strcmp(nums[k], del)==0){ // =====================

if(j>=2){ // идентично (\*)

stack[j-2]=stack[j-2]/stack[j-1]; //

stack[j-1]=0; //

j-=2; //

k++; //

} //

else{ //

m=1; //

j=i; //

} //

} // =====================

}

} //

for(j=1;j<i;j++){ // проверка наличия

if(stack[j]!=0){ // лишних элементов

m=1; // в стеке

} //

} //

if(m==0){ //

printf("\n%d", stack[0]); // вывод результата

} //

else{ //

printf("error"); //

} //

return 0; // завершение работы программы

}