**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 |  | Берленко Т. А. |

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

Функция разбивает массив на токены через пробел с помощью strtok()

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

while(token!=NULL)

{

int numb=strlen(token);

strncpy(nums[i], token, numb);

token=strtok(NULL, " ");

i++;

}

* Перевод символов в числа

Функция переводит символы в числа типа int если они не совпадают со знаками операций

if((strcmp(nums[j], plus)!=0)&&(strcmp(nums[j], minus)!=0)&&(strcmp(nums[j], umn)!=0)&&(strcmp(nums[j], del)!=0)){

stack[k]=atoi(nums[j]);

k++;

}

* Выявление символов операций (в данном случае сложения)

Функция сравнивает элемент массива и знак операции с помощью strcmp()

else if(strcmp(nums[j], plus)==0){

if(k>=2){

pushplus(stack, stack, k-2, k-1);

k-=1;

}

else{

m=1;

j=i;

}

}

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>

#include <string.h>

void pushnull(int\* stack, int j)

{

stack[j]=0;

}

void push(int\* stack, int\* old, int j, int k)

{

stack[j]=old[k];

pushnull(stack, j+1);

}

void pushplus(int\* stack, int\* old, int j, int k)

{

stack[j]+=old[k];

pushnull(stack, j+1);

}

void pushminus(int\* stack, int\* old, int j, int k)

{

stack[j]-=old[k];

pushnull(stack, j+1);

}

void pushumn(int\* stack, int\* old, int j, int k)

{

stack[j]\*=old[k];

pushnull(stack, j+1);

}

void pushdel(int\* stack, int\* old, int j, int k)

{

stack[j]/=old[k];

pushnull(stack, j+1);

}

int top(int\* stack, int j)

{

return stack[j];

}

int size(int\* stack, int I)

{

int j=0;

int k=0;

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

if(stack[j]!=0)

k++;

}

return k;

}

int main() {

int i=0;

int j=0;

int k=0;

int m=0;

char plus[2]="+";

char minus[2]="-";

char umn[2]="\*";

char del[2]="/";

char origin[100];

char nums[100][100];

int inums[100];

int stack[100];

fgets(origin, 100, stdin);

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

while(token!=NULL)

{

int numb=strlen(token);

strncpy(nums[i], token, numb);

token=strtok(NULL, " ");

i++;

}

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

inums[j]=atoi(nums[j]);

}

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

if(top(inums, k)!=0){

push(stack, inums, j, k);

k++;

}

else{

if(strcmp(nums[k], plus)==0){

if(j>=2){

pushplus(stack, stack, j-2, j-1);

j-=2;

k++;

}

else{

m=1;

j=i;

}

}

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

if(j>=2){

pushminus(stack, stack, j-2, j-1);

j-=2;

k++;

}

else{

m=1;

j=i;

}

}

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

if(j>=2){

pushumn(stack, stack, j-2, j-1);

j-=2;

k++;

}

else{

m=1;

j=i;

}

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

if(top(stack, j-1)!=0){

if(j>=2){

pushdel(stack, stack, j-2, j-1);

j-=2;

k++;

}

else{

m=1;

j=i;

}

}

else{

push(stack, inums, j, k);

k++;

}

}

}

if(size(stack, i)!=1)

m=1;

if(m==0){

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

}

else{

printf("error");

}

return 0;

}