**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №2**

**по дисциплине «Программирование»**

Тема: Моделирование стека

|  |  |  |
| --- | --- | --- |
| Студент гр. 6304 |  | Запевалов А.И. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2017

**Цель работы**

Необходимо реализовать программу, которая моделирует работу стека на базе массива.

**Задание**

Моделирование стека.

Требуется написать программу, моделирующую работу стека, реализовав перечисленные ниже методы. Программе на вход подается последовательность команд с новой строки (не более 100 команд), в зависимости от которых программа выполняет ту или иную операцию и выводит результат ее выполнения с новой строки.

Перечень команд:

push n - добавляет целое число n в стек. Программа должна вывести "ok"

pop - удаляет из стека последний элемент и выводит его значение на экран

top - программа должна вывести верхний элемент стека на экран не удаляя его из стека

size - программа должна вывести количество элементов в стеке

exit - программа должна вывести "bye" и завершить работу

Если в процессе вычисления возникает ошибка (например вызов метода pop при пустом стеке), программа должна вывести "error" и завершиться.

Стек требуется реализовать самостоятельно на базе массива.

Рассмотрим основные функции программы.

Введенные команды разделяются на лексемы пробелами и сохраняются в массиве строк.

char \*p=strtok(vvod[n], del); // Divides words from text into strings

while (p != NULL)

{

strcpy(mas[i], p);

p = strtok (NULL, del);

i++;

}

Введенные числа попадают в массив, находящийся в структуре:

typedef struct Stack{

int mas[200];

int size;

} Stack;

Выполняются действия в зависимости от введенных команд: вывод размера массива, вывод последнего элемента массива, добавление элемента увеличивает размер стека на 1, удаление уменьшает на 1.

Исходный код программы:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct Stack{

int mas[200];

int size;

} Stack;

void push(Stack \*stack, char\* a)

{

stack->mas[stack->size]=atoi(a);

printf("ok\n");

stack->size++;

}

void size(Stack \*stack)

{

printf("%d\n", stack->size);

}

int printexit()

{

printf("bye\n");

return 1;

}

void pop(Stack \*stack)

{

printf("%d\n", stack->mas[stack->size-1]);

stack->size--;

}

void top(Stack \*stack)

{

printf("%d\n", stack->mas[stack->size-1]);

}

int main()

{

Stack stack;

int i=0, c=1, n, j=0, flag=0;

char del[]="\n ";

char vvod[200][20];

char mas[200][20];

vvod[0][0]='0';

while(strcmp(vvod[c-1], "exit\n")!=0)

{

fgets(vvod[c], 20, stdin);

//printf("%s\n", vvod[c]);

c++;

}

for (n=1; n<c; n++)

{

char \*p=strtok(vvod[n], del);

while (p != NULL)

{

strcpy(mas[i], p);

p = strtok (NULL, del);

i++;

}

}

for (n=0; n<i; n++)

{

if (flag==0)

{

if (strcmp(mas[n], "push")==0)

{

push(&stack, mas[n+1]);

n++; //jumps over the number

}

else if (strcmp(mas[n], "size")==0)

{

size(&stack);

}

else if (strcmp(mas[n], "exit")==0)

{

flag=printexit();

}

else if (stack.size!=0)

{

if (strcmp(mas[n], "pop")==0)

{

pop(&stack);

}

else if (strcmp(mas[n], "top")==0)

{

top(&stack);

}

}

else

{

printf("error\n");

flag=1;

}

}

}

return 0;

}

**Вывод.**

В данной работе была разработана программа, которая позволяет создавать и обрабатывать массивы подобно стеку – вносить и удалять элементы, считывать их.