**Отчет по лабораторной работе №11**

**Тема**

**Стек**

*РК6-16Б*

16. Петраков Станислав Альбертович

Преподаватель: Федорук Елена Владимировна

**Цель работы**

Применение функций управления динамической памятью для организации динамического стека.

**Ответы на контрольные вопросы**

1. *Что такое стек? Как он организуется?*

Одна из разновидностей линейно динамической структуры. В стеке можно добавлять элемент только вконец; удалять только последний.

1. *Что такое структура?*

Структура — это набор элементов, которые могут иметь различные типы.

1. *Чем структура отличается от массива?*

В массиве переменные одного типа, в структуре могут быть несколько разных переменных с разными типами данных.

1. *Что такое шаблон структуры?*

Шаблон структуры представляет собой список объявлений переменных, описывающий элементы структуры.

1. *Как можно объявить структуру?*

struct [имя\_шаблона] {список\_элементов};

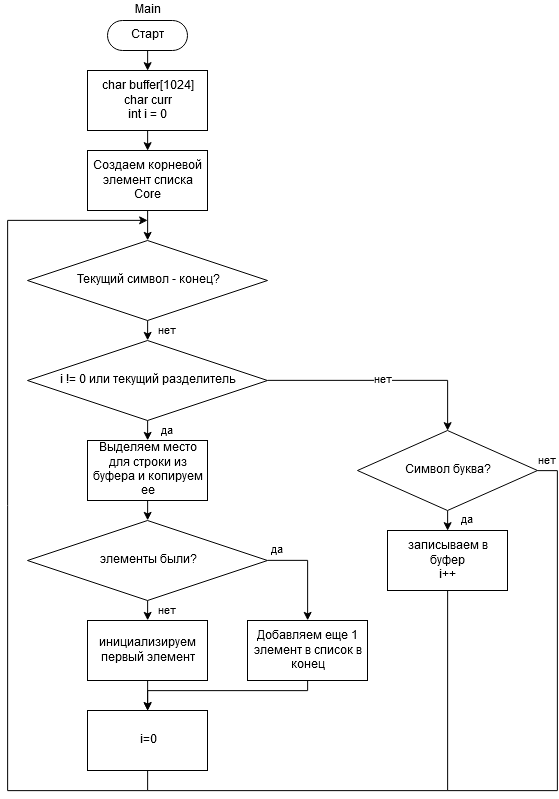
1. *Как организуется линейный односвязный список?*

В элементе хранятся данные и указатель на следующий элемент списка, в последнем элементе хранится NULL указатель.

**Программа**

*Задание:* *Разработайте программу, которая читает из стандартного потока ввода слова и размещает их в стек. В данном случае под словом понимается совокупность видимых символов, расположенных между символами-разделителями. Стек реализуется на основе однонаправленного списка. Перед добавлением в стек из слова удаляются все символы-не-буквы. После окончания ввода содержимое списка выводится в стандартный поток вывода.*

**Блок-схема:**



**Сама программа:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define BUF\_SIZE 100

typedef struct element

{

char\* word;

struct element\* next;

}element\_s;

element\_s\* initElement(char\* str)

{

element\_s\* first;

if ((first = (element\_s\*)malloc(sizeof(element\_s))) == NULL)

{

printf("Memory allocation error. Terminate.\n");

exit(2);

}

first->word = str;

first->next = NULL;

return first;

}

element\_s\* addElement(element\_s\* last, char\* str)

{

element\_s\* temp;

if ((temp = (element\_s\*)malloc(sizeof(element\_s))) == NULL)

{

printf("Memory allocation error. Terminate.\n");

exit(2);

}

temp->word = str;

temp->next = NULL;

return temp;

}

void output(element\_s\* first)

{

for (; first != NULL; first = first->next)

printf("%s\n", first->word);

}

int main()

{

//variables for buffer

char buffer[BUF\_SIZE + 1], curr;

int i = 0;

//variables for list

element\_s\* Core\_Element = NULL;

while ((curr = getchar()) != EOF && curr != '~')

{

if (i != 0 && (curr == ' ' || curr == '\t' || curr == '\n'))

{

char\* temp;

buffer[i] = '\0';

if ((temp = (char\*)malloc(strlen(buffer) \* sizeof(char))) == NULL)

{

printf("Memory allocation error. Terminate.\n");

exit(2);

}

strcpy(temp, buffer);

if (Core\_Element == NULL)

Core\_Element = initElement(temp);

else

{

element\_s\* last = Core\_Element;

while (last->next != NULL)

last = last->next;

last->next = addElement(last, temp);

}

i = 0;

}

else

{

if ((curr >= 'a' && curr <= 'z') || (curr >= 'A' && curr <= 'Z'))

{

//is Overflowed buffer?

if (i == BUF\_SIZE)

{

printf("Buffer is overflowed. Terminate.\n");

exit(1);

}

buffer[i] = curr;

i++;

}

}

prev = curr;

}

output(Core\_Element);

return 0;

}

**Результаты работы программы**

qweewqr wqe67%^$(!reugw

8wllxd dfrslekx903ljdf fdsjkh

~

qweewqr

wqereugw

wllxd

dfrslekxljdf

fdsjkh

**Вывод**

Научился работать со стеком.