## Практические задания для Урока 12 Модуля 2

1. Напишите программу для хранения информации о книгах (автор, количество) в стеке. Программа должна запрашивать ввод информации о пяти книгах с клавиатуры и сохранять в стеке. Напишите функцию удаления элемента стека. Программа должна вывести содержимое стека и вершину стека, спросить, сколько элементов необходимо удалить, удалить указанное количество элементов из стека и вывести содержимое стека и вершину стека.

Очистить память и вывести содержимое стека.

Пример ввода и вывода:

Введите автора: Пушкин

Введите количество: 1

Введите автора: Лермонтов

Введите количество: 2

Введите автора: Блок

Введите количество: 3

Введите автора: Есенин

Введите количество: 4

Введите автора: Бунин

Введите количество: 5

==================

Бунин 5

Есенин 4

Блок 3

Лермонтов 2

Пушкин 1

Вершина стека: Бунин 5

Сколько элементов удалить? 3

==================

Лермонтов 2

Пушкин 1

Вершина стека: Лермонтов 2

==================

1. Напишите программу для создания и вывода на экран списка и количество книг. Создайте структуру для хранения информации о книге (автор, количество). Создайте структуру “список” с раздельным хранением данных и ссылки на следующий элемент списка. Программа должна запрашивать информацию о 5 книгах и каждый раз добавлять элемент в конец списка. Информация вводится с клавиатуры в формате:

Автор

Количество

Программа должна вывести на экран содержимое списка в формате:

Автор:sss Количество:dd

Например:

Автор:Пушкин Количество:10

Автор:Гоголь Количество:20

Автор:Бунин Количество:50

Автор:Блок Количество:10

Автор:Есенин Количество:40

## Решения

1.

#include <stddef.h>

#include <stdlib.h>

#include <malloc.h>

typedef struct

{ // шаблон данных элемента стека

char name[80];

int kol;

} Info;

struct Node

{ // шаблон элемента стека

Info data;

struct Node \*next;

};

struct Node \*head; // вершина стека

// добавление элемента в стек

struct Node \*addStack(Info add)

{

struct Node \*p;

p = (struct Node \*)malloc(sizeof(struct Node));

if (p)

{ // память выделена успешно

p->data = add; // заполнение данных

p->next = head; // ссылка на предыдущий элемент стека

head = p; // вставка в вершину стека

}

return p; // возврат адреса нового элемента или NULL – ошибка добавления

}

// удаление элемента из стека

void delStack()

{

struct Node \*p;

if (head != NULL)

{ // стек не пустой

p = head->next; // запомнить адрес следующего

free(head); // освобождение памяти

head = p; // вершина указывает на новый элемент

}

}

// удаление всех элементов из стека (очистка стека)

// в конце работы функции: head = NULL

void freeStack()

{

struct Node \*p;

while (head != NULL)

{ // пока в стеке есть элементы

p = head->next;

free(head);

head = p;

}

}

// печать содержимого стека

void printStack()

{

struct Node \*p;

printf("==================\n");

p = head; // стать в вершину стека

while (p != NULL)

{

printf("%s %d\n", p->data.name, p->data.kol);

p = p->next; // переход к следующему элементу

}

}

int main()

{

int i, n = 5;

Info s;

head = NULL; // стек пустой

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

{

printf("Введите автора: ");

scanf("%s", s.name);

printf("Введите количество: ");

scanf("%d",&s.kol);

if (!addStack(s))

{

printf("Ошибка добавления в стек!\n");

break;

}

}

printStack(); // в стеке 5 элементов

printf("Вершина стека: %s %d\n",head->data.name, head->data.kol);

printf("Сколько элементов удалить? ");

int n1=0;

scanf("%d",&n1);

for (int i=0; i<n1; i++)

delStack();

printStack();

printf("Вершина стека: %s %d\n",head->data.name, head->data.kol);

freeStack();

printStack(); // в стеке 0

return 0;

}

2.

#include <stdlib.h>

#include <malloc.h>

typedef struct

{ // шаблон данных элемента списка

char name[20]; // название книги

int kol; // количество книг

} InfoBook;

struct List

{ // шаблон элемента списка

InfoBook data;

struct List \*next;

};

struct List \*head; // глобальный указатель – голова списка

// вставка нового элемента в конец списка

struct List \*addEndList(InfoBook a)

{

struct List \*add, \*p;

add = (struct List \*)malloc(sizeof(struct List));

if (add)

{ // память выделилась

add->data = a; // информационная часть

add->next = NULL; // новый элемент будет последним

if (head == NULL) // в списке нет элементов

head = add; // вставка первого элемента

else

{

p = head; // становимся на начало списка

while(p->next != NULL) // пока не последний элемент

p = p->next; // переходим к следующему элементу

p->next = add; // вставка нового следом за последним

}

}

return add; // адрес нового элемента или NULL

}

int main()

{

InfoBook a;

int i, n = 5;

struct List \*p;

// создание списка из 5 элементов

head = NULL;

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

{

printf("Введите фамилию автора: ");

scanf("%s", a.name);

printf("Введите количество: ");

scanf("%d",&a.kol);

if (!addEndList(a))

{ // addEndList(a) == NULL

printf("Ошибка создания списка\n"); break;

}

}

// печать всего списка

p = head; // стать на первый элемент списка

while (p != NULL)

{ // пока не дошли до конца списка

printf( "Автор:%s Количество:%d\n", p->data.name, p->data.kol );

p = p->next; // перейти к следующему элементу списка

}

// удаление списка с освобождением всей занятой им памяти

while(head != NULL)

{

p = head->next; free(head); head = p;

}

return 0;

}