ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ #3

за курсом «Алгоритмы и структуры данных»

студента группы ПА-19-2

Ильяшенко Егора Виталиевича

кафедра компьютерных технологий, ДНУ

2019/2020 уч. Год

Односвязные списки

Вариант №8

1. Постановка задачи:

Реализовать односвязный список с основными операциями. Выполнить задачу по варианту.

1. Описание решение

Разбил задачи на функции, которые вызываются с меню. Функции с задачами вызываются через Switch-case по соответствующему номеру задачи. Реализовал односвязный список с основными его операциями.

1. Исходный текст программы

#include <iostream>

using namespace std;

//variant 8

void Menu();

void AddElem();

void RMElem();

void ShowElem();

struct Sll //Singly linked list

{

int elem;

Sll\* next;

};

Sll \*sll;

void AddElem(int elem\_)

{

Sll\* x = new Sll;

x->elem = elem\_;

x->next = sll;

Sll\* buff = x;

x = sll;

sll = buff;

}

void RMElem(int ind)

{

Sll\* buff = sll;

Sll\* prev = NULL;

for (int i = 1; i != ind;)

{

if (buff->next != NULL)

{

prev = buff;

buff = buff->next;

i++;

}

else

{

cout << "Такого элемента не существует" << endl;

system("Pause");

break;

}

}

if (prev == NULL)

{

buff = sll->next;

delete(sll);

sll = buff;

}

else if (buff->next == NULL)

{

delete(buff);

prev->next = NULL;

}

else

{

prev->next = buff->next;

delete(buff);

}

}

void Task()

{

ShowElem();

int count = 0;

Sll\* buff = sll;

for (int i = 1; buff->next!=NULL ; i++)

{

buff = buff->next;

if (i % 3 == 0)

{

RMElem(i-count);

count++;

}

}

cout << endl << endl;

ShowElem();

}

void ShowElem()

{

Sll\* buff = sll;

for (int i = 0; buff->next != NULL; i++)

{

cout << buff->elem << endl;

buff = buff->next;

}

}

void RMSll()

{

Sll\* buff1 = sll->next;

Sll\* buff2 = buff1->next;

if (buff1 != NULL)

for (; buff2->next != NULL;)

{

delete(buff1);

buff1 = buff2;

buff2 = buff2->next;

}

delete(buff2);

sll->next = NULL;

}

int main()

{

setlocale(LC\_ALL, "Russian");

sll = new Sll;

sll->next = NULL;

Menu();

system("Pause");

}

void Menu()

{

bool menu\_i = true;

while (menu\_i)

{

system("cls");

cout << "1) Добавить элемент" << endl;

cout << "2) Удалить элемент" << endl;

cout << "3) Вывести элементы на экран" << endl;

cout << "4) Очистить список" << endl;

cout << "5) Инд. задание" << endl;

cout << "0) Выход" << endl;

cout << endl;

cout << "Введите номер пункта, к которому хотите перейти - ";

int n\_; cin >> n\_; cout << endl;

switch (n\_)

{

default:

menu\_i = 0;

break;

case 1:

cout << "Введите элемент (int): ";

int el; cin >> el;

AddElem(el);

system("Pause");

break;

case 2:

cout << "Введите номер элемента: ";

int ind; cin >> ind;

RMElem(ind);

system("Pause");

break;

case 3:

ShowElem();

system("Pause");

break;

case 4:

RMSll();

system("Pause");

break;

case 5:

Task();

system("Pause");

break;

case 0:

menu\_i = 0;

system("Pause");

break;

}

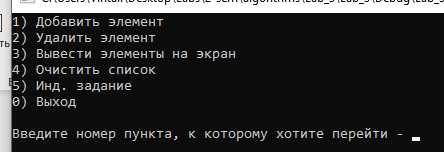
}

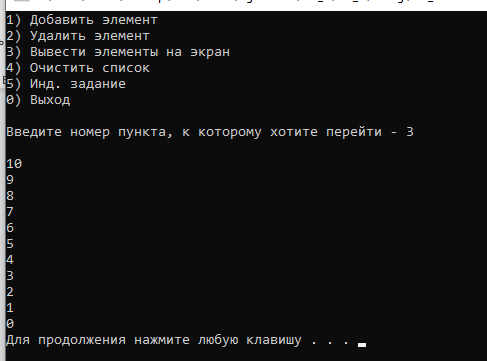
1. Описание интерфейса

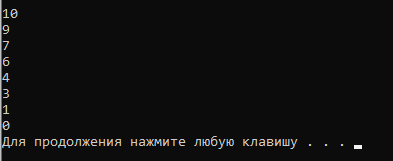
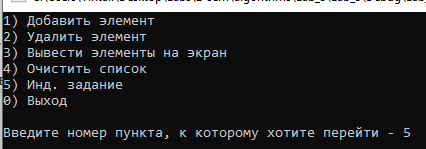
При запуске программы выводится меню. Чтобы перейти к пункту меню, необходимо ввести номер пункта с клавиатуры и нажать ENTER. Дальше следовать инструкциям.

1. Описание тестовых примеров

Вводил разные значения, после чего добавлял ограничения на вводимые данные. Если данные не соответсвовали требуемым, консоль очищалась, выводилось сообщение об ошибке ввода. После этого подпрограмма перезапускается.







Выводы: Структура данных имеет специфическое назначение. Поиск элементов в такой структуре имеет линейную сложность. Преемущество заключается в быстрой манипуляции элементами (удалить, добавить элемент в определённом месте).