Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

Пермский национальный исследовательский политехнический университет

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

**ОТЧЁТ**

**по дисциплине «Информатика»**

Семестр: 2

На тему: «Информационные динамические структуры»

Выполнил студент ИВТ-22-2б:

Зайцев Павел Денисович

Проверил доцент кафедры ИТАС:

Полякова Ольга Андреевна

Пермь 2023

**Постановка задачи**

Написать программу, в которой создаются динамические структуры и выполнить их обработку в соответствии со своим вариантом.

**Вариант**

Записи в стуктуре содержат ключевое поле типа int. Сформировать стек. Удалить из него К элементов перед элементом с заданным номером, добавить К элементов в конец стека.

**Функции**

1. Функция make\_stek() создаёт очередь.
2. Функция cout\_stek() выводит в консоль стек.
3. Функция udalit() удаляет К элементов от К-ого элемента стека.
4. Функция dobav\_stek() добавляет элементы в конец стека.
5. Функция unich() удаляет список очередь.
6. Функция file1() записывает стек в файл.
7. Функция file2() считывает стек с файла.

**Анализ задачи**

1. Стек это структура, каждый элемент которой,называемой узлом, связан с последующим элементом и предыдущим.
2. Удалять и добавлять узлы в стек можно только с его конца.
3. Объявить глобальную переменную счётчик элементов в списке.
4. Реализовать функцию меню для непрерывной работы программы.
5. Объявить глобальные указатель на конец стека.

**Код программы**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <windows.h>

using namespace std;

int kol = 0;//Счетчик количества узлов

struct stek {

int key;

stek\* left;

stek\* right;

};

stek\* kon = NULL;//Глобальный указатель конца основного стека

void make\_stek(int n) {

int t = 1;//счетчик элементов стека

stek\* p = new(stek);//Инициализация первого узла стека в буферный указатль

p->left = NULL;

p->right = NULL;

cout << "\nВведите " << t << "-ый узел стека:";

cin >> p->key;//Ввод ключа i-го списка

kon = p;

t++;

for (int i = n; i > 1; i--) {//Инициализация следующих узлов стека

stek \* p = new(stek);

cout << "\nВведите " << t << "-ый узел стека:";

cin >> p->key;//Ввод ключа i-го списка

p->right = kon;

p->left = NULL;

kon->left = p;

kon = p;

t++;

}

}

void cout\_stek() {

stek\* mm = kon;//Указатель на конец стека

stek\* jj = kon;//Указатель на конец стека

int i = 1;//Инициализация счетчика для вывода номера узла списка

bool flag = true;

if (kon == NULL) {//Выход из функции, если передается указатель без списка

cout << "\nСтека не существует!!!\n";

return;

}

else {

while ((mm->right) != NULL) {//Проверка списка на заполненность

if ((mm->key) != 0)flag = false;

mm = mm->right;

}

if (((mm->right) == NULL) && ((mm->left) == NULL) && ((mm->key) != 0)) {//Проверка списка на заполненность, если список состоит из 1 узла

flag = false;

}

if (flag == false) {//Вывод списка

cout << "\nНаш стек: NULL";

while ((jj->right) != NULL) {

cout << " <- " << jj->key;

jj = jj->right;

i++;

}

cout << " <- " << jj->key << " ]\n";

}

else {

cout << "\nСтек пуст!!!\n";

}

}

}

void udalit() {

int hh, nn;

stek \* kk = kon;//Инициализация указателя на первый узел списка

stek \* f, \* G, \* D;//Инициализация указателей для выполнения удаления

if (kk == NULL) {//Проверка на наполненность передоваемого указателя

cout << "\n!!!ОШИБКА!!!\n";

return;

}

if (((kk->right) == NULL)) {//Проверка присутствия файла

cout << "\nОперация недоступна!!!!\n";

return;

}

cout << endl;

cout << "Введите № узла от которого хотите удалять перед ним стоящие узлы (от 2-" << kol << "):";

cin >> hh;

cout << "Введите количество узлов которые хотите удалить (от 1-" << hh - 1 << "):";

cin >> nn;

int x = 0;

if (nn == hh - 1) {

f = new(stek);//Создание первого узла нового стека

f->left = NULL;

f->right = NULL;

f->key = kk->key;

G = f;

kk = kk->right;

for (int i = kol; i > hh; i--) {//Создание следующих узлов нового стека

f = new(stek);

f->left = NULL;

f->right = G;

f->key = kk->key;

G = f;

kk = kk->right;

}

while (kk->right != NULL) {//Удаление узлов из первого стека

f = kk;

kk=kk->right;

delete f;

}

delete kk;

if ((G->right != NULL) && (G->left == NULL)) {//Переписывание стека состоящего не из одного узла

f = new(stek);//Буферный указатель на первый узел стека

f->left = NULL;

f->right = NULL;

f->key = G->key;

kon = f;//Переприсваивание указателя конца стека

G = G->right;

while (G->right != NULL) {

f = new(stek);

f->left = NULL;

f->right = kon;

f->key = G->key;

kon = f;

G = G->right;

}

f = new(stek);

f->left = NULL;

f->right = kon;

f->key = G->key;

kon = f;

}

else {//Переписывание стека состоящего из одного узла

kon = G;

}

}

else {

f = new(stek);//Создание первого узла нового стека

f->left = NULL;

f->right = NULL;

f->key = kk->key;

G = f;

kk = kk->right;

for (int i = kol; i > hh; i--) {//Создание следующих узлов нового стека

f = new(stek);

f->left = NULL;

f->right = G;

f->key = kk->key;

G = f;

kk = kk->right;

}

for (int y = 1; y <= nn; y++) {//Удаление узлов из первого стека

f = kk;

kk = kk->right;

delete f;

}

kon = kk;//Переприсваивание указателя конца стека

while (G->right != NULL) {//Переписывание нового стека в старый

f = G;

G = G->right;

kon->left = f;

f->right = kon;

f->left = NULL;

kon = f;

}

f = G;

kon->left = f;

f->right = kon;

f->left = NULL;

kon = f;

}

kol -= nn;

}

void dobav\_stek() {

if (kon == NULL) {//Проверка на наполненность передоваемого указателя

cout << "\n!!!ОШИБКА!!!\n";

return;

}

cout << "\nВведите количество узлов,которые вы хотите добавить в конец стека:";

int vv;

int d = 1;

cin >> vv;

if (vv < 1) {

cout << "Неправильный ввод. Нужно повторить операцию\n";

dobav\_stek();//Рекурсивный вызов функции при неправильном вводе

}

kol += vv;

for (int r = 0; r < vv; r++) {

stek\* hh = new(stek);

cout << "\nВведите " << d << "-ый добавляемый элемент очереди:";

cin >> hh->key;//Ввод ключа i-го списка

hh->right = kon;

hh->left = NULL;

kon->left = hh;

kon = hh;

d++;

}

}

void unich() {

if (kon == NULL) {//Проверка на наполненность передоваемого указателя

cout << "\n!!!ОШИБКА!!!\n";

return;

}

while (kon->right != NULL) {//Удаление узлов списка из памяти

stek\* mm = kon->right;

delete kon;

kon = mm;

}

delete kon;//Удаление последнего узла списка из памяти

kon = NULL;

return;

}

void file1() {

int f = 0;//Буферная переменная

FILE\* kk;//Ссылка на файл

stek\* point = kon;

if (kon == NULL) {//Проверка на наполненность передоваемого указателя

cout << "\n!!!ОШИБКА!!!\n";

return;

}

if ((kk = fopen("123.txt", "w")) == NULL) {//Проверка наличия файла

cout << "Произошла ошибка\n";

exit(1);

}

while (point->right != NULL) {//Запись узлов очереди в файл

f = point->key;

fwrite(&f, sizeof(int), 1, kk);

point = point->right;

}

f = point->key;//Запись последнегго узла списка в файл

fwrite(&f, sizeof(int), 1, kk);

fclose(kk);

}

void file2() {

kol = 0;

int g;//Буферная переменная для чтения узлов с файла

FILE\* G;

if ((G = fopen("123.txt", "r")) == NULL) {

cout << "Произошла ошибка\n";

exit(1);

}

fread(&g, sizeof(int), 1, G);

stek\* p = new(stek);//Содание буферного указателя на превый узел стека

p->right = NULL;

p->left = NULL;

p->key = g;

kon = p;

kol++;

while (fread(&g, sizeof(int), 1, G)) {

stek\* m = new(stek);//Содание буферного указателя на последующие узлы стека

m->right = NULL;

m->left = p;

m->key = g;

p->right = m;

p = m;

kol++;

}

cout << "\nEсли до 8-ой команды вы работали с другим списком, то он будет утерян.\n";

return;

}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int n = 0;

int h;

while (true) {

cout << "\nВведите номер операции.\n1)Cоздать стек\n2)Вывести очередь\n3)Удалить К элементов\n4)Вставить К элементов\n5)Удалить стек\n6)Сохранить стек в файл\n7)Достать стек из файла\n8)Закрыть программу\n №=";

cin >> n;

switch (n) {

case 1:cout << "\nВведите количество элементов в стеке:"; cin >> h; kol = h; make\_stek(h); break;

case 2: cout\_stek(); break;

case 3: udalit(); ; break;

case 4: dobav\_stek(); break;

case 5: unich(); break;

case 6: file1(); cout << "7+\n"; break;

case 7: file2(); break;

case 8: return 0;

default: cout << "\n!!Неправильный ввод!!\n";

}

}

}

**Блок схема**

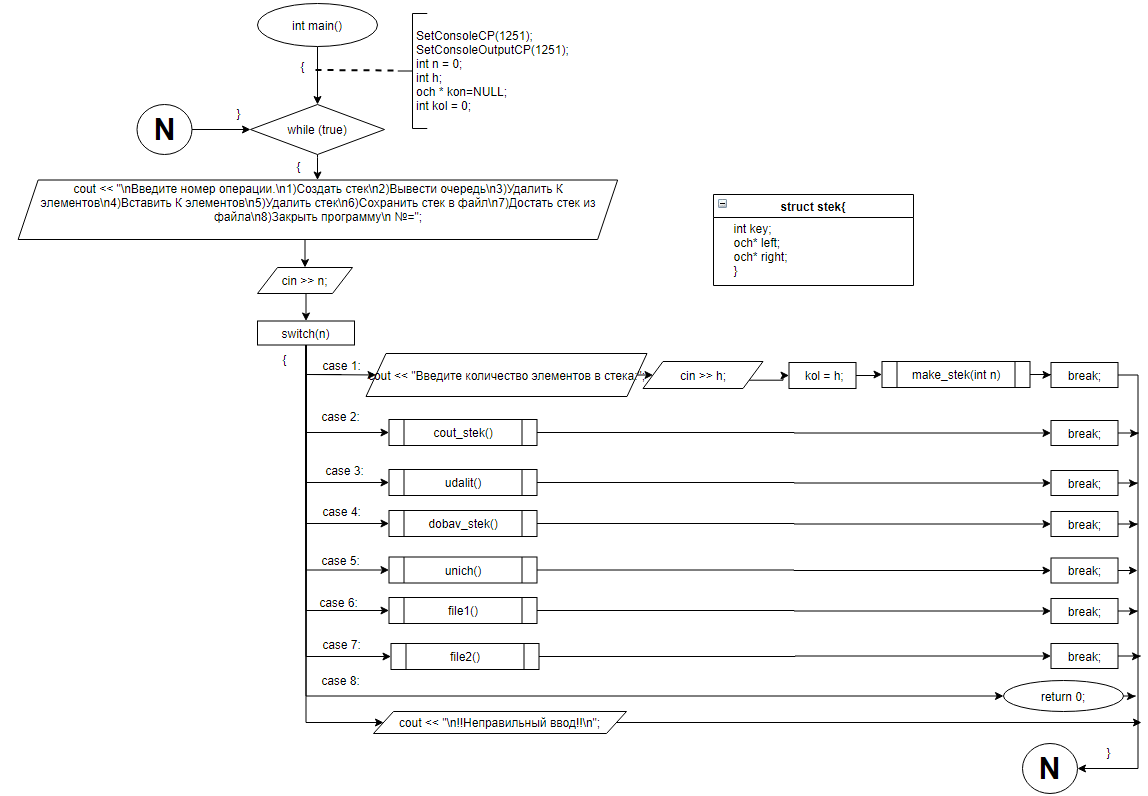


Рисунок 1 -Блок схема main() функции

**Вывод программы**

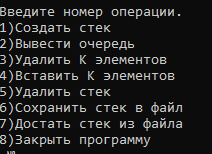


Рисунок 2-Вывод меню программы

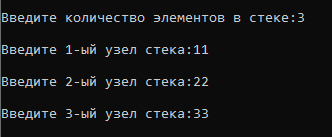


Рисунок 3-Вывод команды №1 программы



Рисунок 4-Вывод команды №2 программы



Рисунок 5-Вывод команды №3 программы

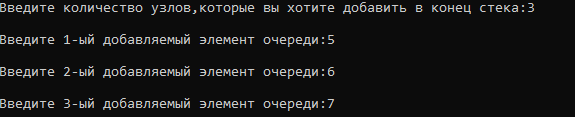


Рисунок 6-Вывод команды №4 программы