ГУО «БГУИР»

Факультет инфокоммуникаций

Кафедра защиты информации

Отчет по

Лабораторной работе №2

Динамическая структура СТЕК

Подготовил:

Студент гр.862101

Комса О.С

Проверила:

Семижон Е.А.

Минск 2019

Цель: изучить алгоритмы работы с динамическими структурами данных в виде стека.

Вариант №9.

Написать программу по созданию, добавлению, просмотру и решению поставленной задачи (в рассмотренных примерах это действие отсутствует) для однонаправленного линейного списка типа СТЕК. Реализовать сортировку стека двумя рассмотренными выше методами. В созданном списке определить количество и удалить все элементы, находящиеся между минимальным и максимальным элементами.

Код: #include "pch.h"

#include <iostream>

#include<math.h>

struct Stack {

int info;

Stack \*next;

}\*begin,\*t;

struct Stack1 {

int inf;

Stack1 \*sled;

}\*fir,\*k;

void view1(Stack1\*p) {

Stack1 \*k = p;

if (p == NULL) {

std::cout << " Временный Стек пуст" << std::endl;

}

while (k != NULL) {

std::cout << k->inf << std::endl;

k = k->sled;

}

}

void view(Stack\*p) {

Stack \*t = p;

if (p == NULL) {

std::cout << "Стек пуст" << std::endl;

}

while (t != NULL) {

std::cout << t -> info << std::endl;

t = t->next;

}

}

Stack1\* InStack1(Stack1 \*m, int in) {

Stack1 \*k = new Stack1;

k->inf = in;

k->sled = m;

return k;

}

Stack\* InStack(Stack \*p, int in) {

Stack \*t = new Stack;

t->info = in;

t->next = p;

return t;

}

void delet(Stack \*\*p) {

Stack\*t;

while (\*p != NULL) {

t = \*p;

\*p = (\*p)->next;

delete t;

}

}

void delet1(Stack1 \*\*p) {

Stack1\*m;

while (\*p != NULL) {

m = \*p;

\*p = (\*p)->sled;

delete m;

}

}

void Sort\_info(Stack \*p) {

Stack \*t = NULL, \*t1;

int r;

do {

for (t1 = p; t1->next != t; t1 = t1->next)

if (t1->info > t1->next->info) {

r = t1->info;

t1->info = t1->next->info;

t1->next->info = r;

}

t = t1;

}

while (p->next != t);

}

int main()

{

setlocale(LC\_ALL, "Russian");

int i;

int n = 20;

int in;

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

in = rand() % 86 + (-30);

begin = InStack(begin, in);

}

Sort\_info(begin);

int info = begin->info;

Stack \*s;

for (s = begin; s != NULL; s = s->next)

{

if (s->info > info) {

info = s->info;

}

}

int min = begin->info;

for (s = begin; s != NULL; s = s->next)

{

if (s->info < min) {

min = s->info;

}

}

for (s = begin;s != NULL;s = s->next) {

fir = InStack1(fir, s->info);

}

int kol = 0;

for (s = begin;s != NULL;s = s->next) {

kol++;

}

std::cout << "Стек отсортированный:" << std::endl;

view(begin);

delet(&begin);

std::cout << "Память очищена" << std::endl;

std::cout<<"Максимальный элемент:" << info << " " <<"Минимальный элемент:"<< min << std::endl << std::endl;

std::cout << "Временный стек:" << std::endl;

view1(fir);

std::cout << std::endl << std::endl;

Stack1 \*l;

for (l = fir;l !=NULL;l = l->sled) {

if (l->inf =min ) {

begin = InStack(begin, l->inf);

break;

}

}

for (l = fir;l != NULL;l = l->sled) {

if (l->inf = info) {

begin = InStack(begin, l->inf);

break;

}

}

std::cout << "Конечный Стек:" << std::endl;

view(begin);

delet1(&fir);

std::cout << "Очистка временного стека" << std::endl;

std::cout << "Проверка:" << std::endl;

view1(fir);

std::cout << "Количество элементов стека:" <<kol<< std::endl;

}

Вывод: создал программу для вычисления указанной величины.

