Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования

**Пермский национальный исследовательский политехнический университет**

Факультет Электротехнический Кафедра ИТАС

Специальность Мехатроника и Робототехника

**ОТЧЁТ**

**о лабораторной работе №11**

Информационные динамические структуры

|  |  |
| --- | --- |
| **Пермь 2022** | Выполнил:  Студент группы ПРТ-21-2б  Выймов С.С  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Проверил:  Доцент кафедры ИТАС Поляков О.А.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Постановка задачи:

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

Текст программы:

#include <iostream>

using namespace std;

struct Node

{

int x; //информационный элемент

Node\* Next; //указатель на следующий элемент

};

typedef Node\* PNode;

void Add(int data, PNode& Head, PNode& LastNode) //добавление

{

PNode Temp; // Temp - новый элемент, LastNode - последний

if (Head == NULL)

{

Head = new Node;

LastNode = Head;

Head->Next = NULL;

}

else

{

Temp = new Node;

LastNode->Next = Temp;

LastNode = Temp;

LastNode->Next = NULL;

}

LastNode->x = data;

}

void Show(PNode Head) //отображение очереди

{

while (Head != NULL) //пока текущий элемент не NULL

{

cout << Head->x << " "; //выводим поле

Head = Head->Next; //переходим к следующему элементу

}

}

void DeleteNode(PNode& Head, PNode OldNode)

{

PNode q = Head;

if (Head == OldNode)

Head = OldNode->Next; // удаляем первый элемент

else

{

while (q && q->Next != OldNode) // ищем элемент

q = q->Next;

if (q == NULL)

return; // если не нашли, выход

q->Next = OldNode->Next;

}

delete OldNode; // освобождаем память

}

void AddAfter(int data, PNode p, PNode NewNode)

{

NewNode->Next = p->Next;

NewNode->x = data;

p->Next = NewNode;

}

int main()

{

setlocale(LC\_ALL, "rus");

PNode Head, LastNode;

int n;

int k;

Head = NULL;

LastNode = NULL;

cout << "Введите количество элементов в списке\n";

cin >> k;

cout << "Введите значения\n";

for (int i = 0; i < k; i++) //заносим данные в очередь

{

cin >> n;

Add(n, Head, LastNode);

}

cout << "Список: ";

Show(Head); //выводим очередь

cout << endl;

cout << "Введите номер элемента, который нужно удалить\n";

int x = -1;

while (x > k || x <= 0)

{

cout << "Введите номер от 1 до " << k << endl;

cin >> x;

}

int s = -1;

while (s > k - x + 1 || s <= 0)

{

cout << "Введите сколько удалить от 1 до " << k - x + 1 << endl;

cin >> s;

}

int p = 0;

PNode q = Head;

do

{

PNode q = Head;

for (int i = 0; i < x - 1; i++)

q = q->Next;

DeleteNode(Head, q);

p++;

} while (p < s);

p = 0;

Show(Head); //выводим очередь

cout << "\nВведите номер, под которым нужно добавить элемент\n";

int y = -1;

while (y > k || y <= 0)

{

cout << "Введите номер от 1 до " << k - s + 1 << endl;

cin >> y;

}

cout << "Введите сколько нужно добавить элементов \n";

int h;

cin >> h;

cout << "Введите значение элемента ";

do

{

PNode NewNode = new Node;

int g;

cin >> g;

q = Head;

if (y - 2 < 0)

{

NewNode->Next = Head;

NewNode->x = g;

Head = NewNode;

}

else

{

for (int i = 0; i < y - 2; i++)

q = q->Next;

AddAfter(g, q, NewNode);

}

p++;

} while (p < h);

Show(Head);

}

Результат решения:

