МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

“ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

КАФЕДРА ПРОГРАМНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ УПРАВЛІННЯ

ЗВІТ

З ЛАБОРАТОРНОЇ РОБОТИ №1

ЗА ТЕМОЮ “Базові структури даних”

Виконав студент

групи КН-221д

Габорець

Перевірив

Солонська С.В.

Харків 2021

ЛАБОРАТОРНА РОБОТА №1

***Тема:*** Базові структури даних.

***Мета роботи:*** познайомитися з базовими структурами даних (список, черга, стек) та отримати навички програмування алгоритмів, що їх обробляють.

***Завдання:***

Розробити програму, яка читає з клавіатури послідовність N цілих чисел (1 < N < 256), жодне з яких не повторюється, зберігає їх до структури даних (згідно завданню) та видає на екран наступні характеристики:

кількість елементів;

середнє арифметичне збережених елементів;

мінімальний та максимальний елемент;

четвертий елемент послідовності;

елемент, що йде перед мінімальним елементом.

Наголосимо, що всі характеристики потрібно визначити із заповненої структури даних. Дозволено використовувати лише ті операції, що притаманні заданій структурі, наприклад, заборонено отримувати доступ до елементу із довільною позицією у черзі, яку реалізовано на базі масиву.

Використовувати готові реалізації структур даних (наприклад, STL) заборонено.

Варіант завдання

Використати наступні структури даних.

Двобічно зв’язний список.



Код:

header.h:

#pragma once

#include<iostream>

int validInputChoise(int& choise);

int validInputChoise\_N(int& choise);

struct Node

{

Node\* pNext, \* pPrevious;

int data;

};

class List

{

int size = 0;

public:

Node\* head, \* tail;

List() { head = nullptr; tail = nullptr; };

~List() //Деструктор

{

while (head) //Пока по адресу на начало списка что-то есть

{

tail = head->pNext; //Резервная копия адреса следующего звена списка

delete head; //Очистка памяти от первого звена

head = tail; //Смена адреса начала на адрес следующего элемента

}

};

void pushback(int data);

void full\_print();

int get\_size() { return size; };

double NOD();

void max\_data();

void min\_data();

int take\_elem(int index);

void previous\_min();

};

header.cpp:

#include "header.h"

int validInputChoise\_N(int& choise)

{

while (true)

{

std::cin >> choise;

if (choise < 1 || choise>256) { throw std::exception("1<N<256"); }

if (std::cin.fail())

{

std::cin.clear();

std::cin.ignore(32767, '\n');

std::cout << "Непонятный ввод ";

}

else

{

std::cin.ignore(32767, '\n');

return choise;

}

}

}

int validInputChoise(int& choise)

{

while (true)

{

std::cin >> choise;

if (std::cin.fail())

{

std::cin.clear();

std::cin.ignore(32767, '\n');

std::cout << "Непонятный ввод ";

}

else

{

std::cin.ignore(32767, '\n');

return choise;

}

}

}

void List::pushback(int data)//Запихать в конец

{

Node\* now\_el = head;

while (now\_el != nullptr)

{

if (now\_el->data == data)

{

throw std::exception("Такое число уже есть в списке");

}

else

now\_el = now\_el->pNext;

}

Node\* temp = new Node;

temp->pNext = nullptr;

temp->data = data;

if (head != nullptr)//если елементы есть

{

temp->pPrevious = tail;

tail->pNext = temp;

tail = temp;

}

else //список пустой

{

temp->pPrevious = nullptr;

head = tail = temp;

}

size++;

};

void List::full\_print()

{

Node\* temp = head;

while (temp != nullptr)

{

std::cout << temp->data << "\t";

temp = temp->pNext;

}

std::cout << "\n";

}

double List::NOD()

{

Node\* temp = head;

double summa = 0;

while (temp != nullptr)

{

summa += temp->data;

temp=temp->pNext;

}

return summa;

};

void List::max\_data()

{

Node\* temp = head;

int max = head->data;

while (temp != nullptr)

{

if (max < temp->data)

{

max = temp->data;

}

temp = temp->pNext;

}

std::cout << "\nmax\t" <<max<< std::endl;

};

void List::min\_data()

{

Node\* temp = head;

int min = head->data;

while (temp != nullptr)

{

if (min > temp->data)

{

min = temp->data;

}

temp = temp->pNext;

}

std::cout << "\nmin\t" << min << std::endl;

};

int List::take\_elem(int index)

{

Node\* temp = head;

int i = 0;

while (temp && i < index-1)

{

temp = temp->pNext;

i++;

}

int chislo = temp->data;

return chislo;

}

void List::previous\_min()

{

Node\* temp = head;

Node\* min = head;

while (temp != nullptr)

{

if (min->data > temp->data)

{

min = temp;

}

temp = temp->pNext;

}

if (min->pPrevious == nullptr) std::cout << "\n Число минимума ПЕРВОЕ,перед ним НЕТУ числа..." << std::endl;

min = min->pPrevious;

int previous\_min = min->data;

std::cout << "\nЧисло перед минимумом\t" << previous\_min << std::endl;

};

Source.cpp:

#include <iostream>

#include<Windows.h>

#include"header.h"

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

List lst;

std::cout << "Введите кол-во вводимых вами чисел от 1 до 256\n";

int N;

for(static int i=0;i<1;i++)

try

{

validInputChoise\_N(N);

}

catch (const std::exception& ex)

{

std::cout << ex.what() << std::endl;

i--;

}

for(static int i = 0; i<N;i++)

{

std::cout << "Введите ваше число\n";

int chislo\_data;

validInputChoise(chislo\_data);

try

{

lst.pushback(chislo\_data);

}

catch (const std::exception& ex)

{

std::cout << ex.what() << std::endl;

i--;

}

}

std::cout << "\nвот наш список\n";

lst.full\_print();

std::cout<< "\nРазмер списка\t"<< lst.get\_size() << "\n\n";

std::cout << "Среднее арифметическое\t" << lst.NOD() / lst.get\_size() << std::endl;

lst.max\_data();

lst.min\_data();

std::cout << "\nнаш елемент номер 4\t" << lst.take\_elem(4) << std::endl;

lst.previous\_min();

return 0;

}

***Висновок:*** виконавши лабораторну роботу №1, я отримав навички роботи з базовими структурами даних (список, черга, стек) та отримав навички програмування алгоритмів, що їх обробляють.