МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Череповецкий государственный университет»

**Лабораторная работа № 1**

**«Динамические списки»**

**Выполнил:**

**студент гр.** 1ИВТпб-01-21оп

Климов А.Г. **Проверил: преподаватель**

Пышницкий К.М. **Отметка о зачете:**

Череповец

2017 год

**Ход работы**

**Задание 5.** Даны два стека целых чисел. Сфоpмиpовать очеpедь из элементов пеpвого

стека, кpатных минимуму втоpого.

**Требования к программам:**

1. Количество элементов исходных линейных списков заранее не определено

и задается случайным образом. При дальнейшей обработке считается, что

количество элементов списка не известно, т.е. обработка производится, пока

не достигнут конец списка.

2. Программа должна сформировать исходные линейные списки, вывести их содержимое

на экран (при этом данные из списков не должны быть потеряны), произвести

обработку и вывести содержимое итогового списка на экран.

Выполнить задания, используя для пpедставления очеpедей и стеков:

а) массивы; б) динамические списки.

**Логическое проектирование:**

1. Определение структур стека и очереди;
2. Объявление двух стеков и одной очереди;
3. Задаём максимальный размер стеков и очереди;
4. Генерируем случайные размеры стеков и случайные значения;
5. Вывод значения стеков;
6. Проверка условия и формирование очереди;
7. Вывод полученной очереди.

**Текст программы:**

**а)**

#include "stdafx.h"

#include <iostream>

#include <ctime>

#include <climits>

using namespace std;

int st\_top;

///////////////////////////////////////////////////

////Stack1

///////////////////////////////////////////////////

struct Stack1 {

int top;

int \*data;

};

void InitStack(Stack1 &st1, int capacity) {

st1.data = new int[capacity];

st1.top = -1;

}

void push(Stack1 &st1, int value) {

st1.data[++st1.top] = value;

}

int pop(Stack1 &st1) {

return st1.data[st1.top--];

}

bool empty(Stack1 &st1) {

return st1.top == -1;

}

void print(Stack1 &st1) {

int tmp = st1.top;

while (!empty(st1))

cout << pop(st1) << " ";

st1.top = tmp;

}

void nullStack(Stack1 &st1) {

st1.top = -1;

}

///////////////////////////////////////////////////

////Queue1

///////////////////////////////////////////////////

struct Queue1 {

int head, tail, size;

int \*data;

};

void nullQueue(Queue1 &q) {

q.head = 0; q.tail = q.size - 1;

}

void InitQueue(Queue1 &q, int capacity) {

q.size = capacity + 1;

q.data = new int[q.size];

nullQueue(q);

}

int next\_(Queue1 &q, int n) {

return (n + 1) % q.size;

}

bool empty(Queue1 &q) {

return next\_(q, q.tail) == q.head;

}

void add(Queue1 &q, int value) {

if (next\_(q, next\_(q, q.tail)) == q.head)

cout << "Queue overflow" << endl;

else

{

q.tail = next\_(q, q.tail);

q.data[q.tail] = value;

}

}

int del(Queue1 &q) {

if (empty(q)) {

cout << "Queue is empty" << endl;

return 0;

}

else

{

int d = q.data[q.head];

q.head = next\_(q, q.head);

return d;

}

}

void print(Queue1 &q) {

if (empty(q)) {

cout << "Queue is empty" << endl;

return;

}

else

{

int tmp = q.head;

while (!empty(q)) {

int d = q.data[q.head];

q.head = next\_(q, q.head);

cout << d << " ";

}

q.head = tmp;

}

}

///////////////////////////////////////////////////

////main

///////////////////////////////////////////////////

int main()

{

setlocale(LC\_ALL, "Russian");

Stack1 st1;

Stack1 st2;

Queue1 q;

InitStack(st1, 10);

InitStack(st2, 10);

InitQueue(q, 10);

int min = INT\_MAX;

int tmp1;

srand(time(0)); // автоматическая рандомизация

int random\_value\_st1 = rand() % 10;

int random\_value\_st2 = rand() % 10;

cout << "Размер 1-го стека = " << random\_value\_st1 + 1 << endl; //случайный размер st1

cout << "Размер 2-го стека = " << random\_value\_st2 + 1 << endl; //случайный размер st2

for (int i = 0; i <= random\_value\_st1; i++) { push(st1, 1 + rand() % 10); } //запись случайных чисел в st1

for (int i = 0; i <= random\_value\_st2; i++) { push(st2, 1 + rand() % 10); } //запись случайных чисел в st2

cout << endl;

cout << "1-ый стек:" << endl;

print(st1); //вывод 1-го стека

cout << endl;

cout << "2-ой стек:" << endl;

print(st2); //вывод 2-го стека

cout << endl;

while (!empty(st2)) { tmp1 = pop(st2); if (tmp1 < min) { min = tmp1; } }

while (!empty(st1)) { tmp1 = pop(st1); if (tmp1 % min == 0) { add(q, tmp1); } }

cout << endl;

cout << "Минимум = " << min;

cout << endl;

cout << "Очередь: " << endl;

print(q);

cout << endl;

cout << endl;

system("pause");

return 0;

}

**б)**

#include "stdafx.h"

#include <iostream>

#include <ctime>

#include <climits>

using namespace std;

///////////////////////////////////////////////////

////Stack

///////////////////////////////////////////////////

struct Node1 {

int data; //информационный элемент

Node1 \*next; //Голова стека

};

void InitStack(Node1 \*&top) { top = NULL; }

void push(Node1 \*&top, int value)

{

Node1 \*tmp = new Node1;

tmp->next = top;

top = tmp;

top->data = value;

}

int pop(Node1 \*&top) {

Node1 \*tmp = top;

int d = top->data;

top = top->next;

delete (tmp);

return d;

}

bool empty(Node1 \*&top) {

return top == NULL;

}

void print(Node1 \*top) {

Node1 \*tmp = top; //Объявляем указатель и Указываем ему, что его позиция в голове стека

//с помощью цикла проходим по всему стеку

while (tmp != NULL) //выходим при встрече с пустым полем

{

cout << tmp->data << " "; //Выводим на экран элемент стека

tmp = tmp->next; // Переходим к следующему элементу

}

}

void nullStack(Node1 \*&top) {

Node1 \*tmp;

while (!empty(top)) {

tmp = top;

top = top->next;

delete(tmp);

}

}

///////////////////////////////////////////////////

////Queue

///////////////////////////////////////////////////

class Queue {

private:

struct Node {

int data;

Node \*next;

};

Node \*head, \*tail;

public:

Queue() {

head = NULL;

tail = NULL;

}

bool empty() {

return head == NULL;

}

void add(int value) {

if (empty()) {

head = new Node;

head->data = value;

head->next = NULL;

tail = head;

}

else {

tail->next = new Node;

tail = tail->next;

tail->data = value;

tail->next = NULL;

}

}

int del() {

if (empty()) { cout << "Queue is empty" << endl; return 0; }

else {

int d = head->data;

Node \*tmp = head;

head = head->next;

delete(tmp);

return d;

}

}

void nullQueue() {

Node \*tmp;

while (!empty()) {

tmp = head;

head = head->next;

delete(tmp);

}

}

void print() {

Node \*tmp = head; //Временный указатель на начало списка

while (tmp != NULL) //Пока в списке что-то встречается

{

cout << tmp->data << " "; //Выводим значения из списка на экран

tmp = tmp->next; //Сдвигаем указатель на начало на адрес следующего элемента

}

cout << endl;

}

};

///////////////////////////////////////////////////

////main

///////////////////////////////////////////////////

int main()

{

setlocale(LC\_ALL, "Russian");

Queue q;

Node1 \*s1 = 0;

Node1 \*s2 = 0;

int min = INT\_MAX;

int tmp1;

srand(time(0)); // автоматическая рандомизация

int random\_value\_st1 = rand() % 10;

int random\_value\_st2 = rand() % 10;

cout << "Размер 1-го стека = " << random\_value\_st1 + 1 << endl; //случайный размер st1

cout << "Размер 2-го стека = " << random\_value\_st2 + 1 << endl; //случайный размер st2

for (int i = 0; i <= random\_value\_st1; i++) { push(s1, 1 + rand() % 10); } //запись случайных чисел в st1

for (int i = 0; i <= random\_value\_st2; i++) { push(s2, 1 + rand() % 10); } //запись случайных чисел в st2

cout << endl;

cout << "1-ый стек:" << endl;

print(s1); //вывод 1-го стека

cout << endl;

cout << "2-ой стек:" << endl;

print(s2); //вывод 2-го стека

cout << endl;

while (!empty(s2)) { tmp1 = pop(s2); if (tmp1 < min) { min = tmp1; } }

while (!empty(s1)) { tmp1 = pop(s1); if (tmp1 % min == 0) { q.add(tmp1); } }

cout << endl;

cout << "Минимум = " << min;

cout << endl;

cout << "Очередь: " << endl;

q.print();

cout << endl;

cout << endl;

system("pause");

return 0;

}