**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Лабораторная работа №3**

**по дисциплине «Языки программирования»**

Массивы указателей

Группа: **АВТ-615**

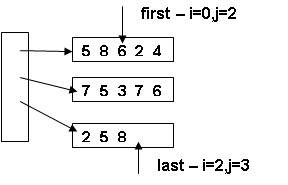
Студент: **Кузенков В.С.**

Преподаватель: **Балагуров М.В.**

НОВОСИБИРСК 2017

# Постановка задачи

Очередь моделируется при помощи динамического массива указателей на линейные массивы размерности N целых. Указатели на первый и последний элементы очереди – два индекса – в массиве указателей и линейном массиве. В операции добавления при переполнении текущего линейного массива в массив указателей добавляется новый, в операции извлечения – при переходе к следующему линейному массиву текущий утилизуется (указатели в массиве указателей смещаются к началу).



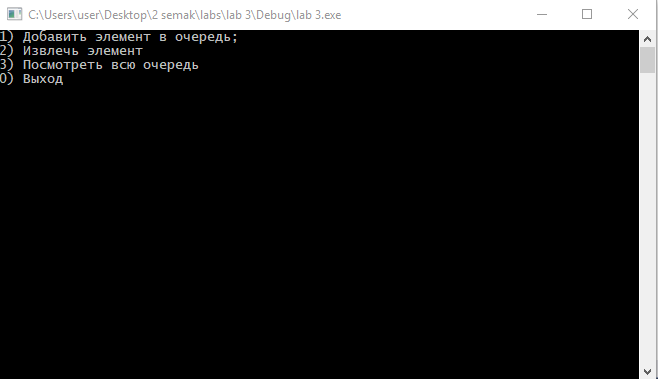
# Ограничения

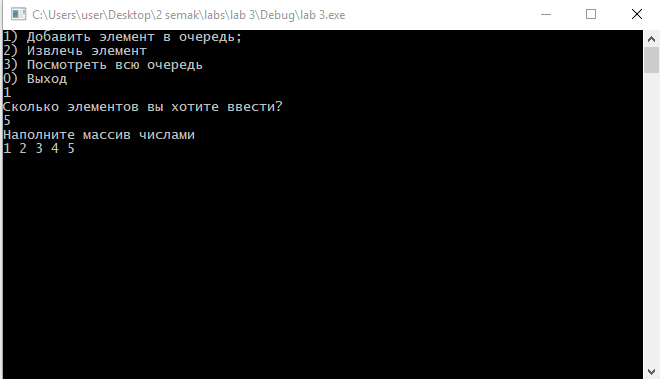
Отсутствуют

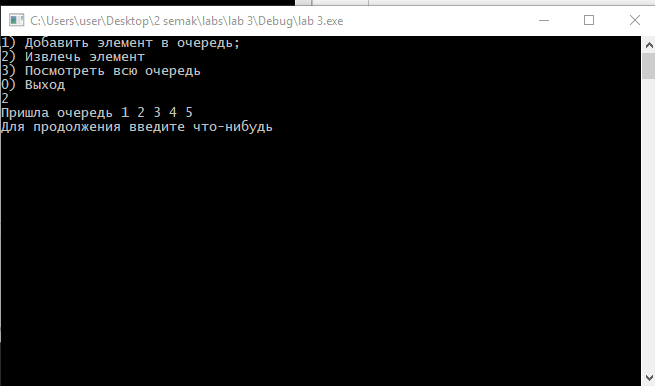
# Тестирование

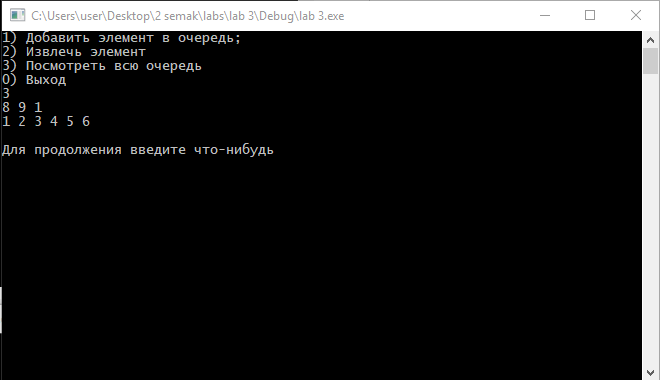
**Входные данные**

**Выходные данные**









# Текст программы:

/\* Очередь моделируется при помощи динамического массива указателей на линейные массивы размерности N целых. Указатели на первый и последний элементы очереди – два индекса – в массиве указателей и линейном массиве. В операции добавления при переполнении текущего линейного массива в массив указателей добавляется новый, в операции извлечения – при переходе к следующему линейному массиву текущий утилизуется (указатели в массиве указателей смещаются к началу).\*/

#include <iostream>

#include <Windows.h>

using namespace std;

static unsigned short N = 10;

unsigned short arr\_count = 3;

int menu(int \*\*queue, int \*queueBeggin, int \*queueEnd);

int q\_relloc(int\*\* queue, bool isNeedMore)

{

int old\_count = arr\_count;

int \*\*temp;

if (isNeedMore)

{

arr\_count++;

temp = static\_cast<int\*\*>(realloc(queue, arr\_count\*sizeof(int\*)));

if (temp == nullptr)

return 1;

queue = temp;

for (int i = arr\_count; i < old\_count; queue[i++] = nullptr); // обнуление новой памяти на всякий

int \*arr = new int[N + 1];

for (int i = 0; i <= N; i++) arr[i] = 0; // заполнение нового места в памяти нулями

queue[arr\_count - 1] = arr;

}

else

{

if(arr\_count == 3)

{

cout << "Очередь кончилась\n";

return 0;

}

arr\_count--;

temp = static\_cast<int\*\*>(realloc(queue, arr\_count\*sizeof(int\*)));

}

return 0;

}

void enqueue(int \*\*queue, int\* queueBeggin, int \*queueEnd) {//добавление элемента(поставить в очередь :3)

int \*arr = new int[N+1];

unsigned int maxSize = 0;

cout << "Сколько элементов вы хотите ввести?\n";

cin >> maxSize;

cout << "Наполните массив числами\n";

unsigned int i = 1;

for (unsigned int counter = 0; counter < maxSize; ++i, ++counter)

{

cin >> arr[i];

if(i%10 == 1 && i > 10)

{

int temp = arr[i];

arr[0] = 10;

queue[arr\_count-1] = arr;

q\_relloc(queue, 1);

arr = new int[N + 1];

i = 1;

arr[i] = temp;

}

if(counter == maxSize-1)

{

arr[0] = maxSize%10;

queue[arr\_count-1] = arr;

q\_relloc(queue, 1);

}

}

queueEnd = &arr[i];

}

int dequeue(int \*\*queue, int \*queueBeggin, int \*queueEnd) {//извлечение элемента

if(queue[2][0] == 0)

{

cout << "Очередь кончилась\n";

int w8;

cout << "Для продолжения введите что-нибудь\n";

cin >> w8;

return 0;

}

cout << "Пришла очередь ";

for (int i = 1; i <= queue[2][0]; ++i)

{

cout << queue[2][i] << " ";

}

cout << endl;

int w8;

cout << "Для продолжения введите что-нибудь\n";

cin >> w8;

delete queue[2];

for (int i = 3; i < arr\_count; ++i)

{

queue[i-1]= queue[i];

}

q\_relloc(queue, 0);

queueBeggin = queue[2];

queueEnd = &queue[arr\_count-1][queue[arr\_count-1][0]-1];

return 1;

}

int show(int \*\*queue, int \*queueBeggin, int \*queueEnd)

{

if (queue[2][0] == 0)

{

cout << "Очередь пустая\n";

int w8;

cout << "Для продолжения введите что-нибудь\n";

cin >> w8;

return 0;

}

for (int i = 2; i < arr\_count; ++i)

{

for (int j = 1; j <= queue[i][0]; ++j)

{

cout << queue[i][j] <<" ";

}

cout << endl;

}

int w8;

cout << "Для продолжения введите что-нибудь\n";

cin >> w8;

return 1;

}

int menu(int \*\*queue, int \*queueBeggin, int \*queueEnd)

{

unsigned short choose;

system("cls");

cout << "1) Добавить элемент в очередь;\n"

<< "2) Извлечь элемент\n"

<< "3) Посмотреть всю очередь\n"

<< "0) Выход\n";

cin >> choose;

switch (choose) {

case(1) : enqueue(queue, queueBeggin, queueEnd); break;

case(2) : dequeue(queue, queueBeggin, queueEnd); break;

case(3) : show(queue, queueBeggin, queueEnd); break;

case(0) : return 0;

default: return menu(queue, queueBeggin, queueEnd);

}

return menu(queue, queueBeggin, queueEnd);

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int \*\*queue = static\_cast<int\*\*>(calloc(arr\_count, arr\_count\*sizeof(int\*)));

int \*queueBeggin = queue[2];

int \*queueEnd = nullptr;

queue[0] = queueBeggin;

queue[1] = queueEnd;

int \*arr = new int[N + 1];

for (int i = 0; i <= N; i++) arr[i] = 0; // заполнение нового места в памяти нулями

queue[arr\_count - 1] = arr;

menu(queue, queueBeggin, queueEnd);

system("pause");

return 0;

}

**Вывод:** Основываясь на практическом опыте было выяснено, что массив указателей (МУ) – простейшая структура данных, в которой проявляется различие между физическим и логическим порядком следования элементов. Способ организации данных ясен уже из самого определения: это массив, каждый элемент которого содержит указатель на переменную (объект).