Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Курский государственный университет»

Кафедра программного

обеспечения и администрирования

информационных систем

Направление подготовки

математическое обеспечение и

администрирование

информационных систем

Форма обучения очная

**Отчет**

**по лабораторной работе №5**

«Очереди»

Выполнил:

студент группы 213 Тихонов Е.Е.

Проверил:

профессор кафедры ПОиАИС Кудинов В.А.

Курск, 2022

**Цель работы**: Научиться составлять процедуры (функции), реализующие операции для работы с очередями.

**Задание**

1) Для работы с очередью напишите процедуру ее формирования, причем учтите, что очередь может содержать не более K звеньев (переполнение очереди) и не может быть пустой (опустошение очереди) (1 балл);

2) Пусть построена очередь Q, содержащая целые числа. Сформировать новую очередь P, состоящую из элементов очереди Q, кратных числу 3 (4 балла);

3) Используя структуру данных очередь, переписать содержимое текстового файла F в текстовый файл G (6 баллов).

4) Напишите процедуру проверки того, имеет ли очередь звено с информационным полем, содержащим целое число A. При формировании очереди учтите, что она может содержать не более K звеньев (переполнение очереди) и не может быть пустой (опустошение очереди) (1 балл).

**Код программы**

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

template <typename DataType>

class Node;

template <typename T>

class Queue

{

public:

friend class Node<T>;

Node<T>\* First;

Node<T>\* Last;

int Count = 0;

int MaxQueueLength = 0;

Queue(T FirstElement, int MaxQueueLength)

{

this->First = new Node<T>(FirstElement);

this->Last = First;

this->MaxQueueLength = MaxQueueLength;

Count++;

}

void PushBack(T Element)

{

if (Count < MaxQueueLength)

{

Node<T>\* temp = new Node<T>(Element);

Last->Next = temp;

temp->Prev = Last;

Last = temp;

Count++;

}

else

{

cout << "переполнение очереди"<<endl;

}

}

T PopFront()

{

Node<T>\* temp = First;

if (Count > 1)

{

First = temp->Next;

Count--;

}

else

{

cout << "опустошение очереди" << endl;

}

return temp->Data;

}

bool IsContains(T Element)

{

Node<T>\* Elem = this->First;

while (Elem != nullptr)

{

if (Elem->Data == Element)

{

return true;

}

Elem = Elem->Next;

}

return false;

}

string ToString()

{

string str1 = "";

Node<T>\* Elem = this->First;

while (Elem != nullptr)

{

str1 += to\_string(Elem->Data);

str1 += " ";

Elem = Elem->Next;

}

return str1;

}

};

template <typename DataType>

class Node

{

friend class Queue<DataType>;

DataType Data;

Node\* Next;

Node\* Prev;

Node(DataType Data = 0)

{

this->Data = Data;

this->Next = nullptr;

this->Prev = nullptr;

}

};

int main()

{

setlocale(LC\_ALL, "Russian");

//3

Queue<int>\* q1 = new Queue<int>(1, 5);

q1->PushBack(3);

q1->PushBack(5);

q1->PushBack(7);

q1->PushBack(9);

cout << "q1: " << q1->ToString() << endl;

Queue<int>\* q2 = new Queue<int>(1, 5);

int count = q1->Count;

for (int i = 0; i < count; i++)

{

int elem = q1->PopFront();

if (elem % 3 == 0)

{

q2->PushBack(elem);

}

}

q2->PopFront();

cout << "q2: " << q2->ToString() << endl;

//5

string Path1 = "C:/Users/Evgeniy/Desktop/Code/Algoritms/task5/text1.txt";

string Path2 = "C:/Users/Evgeniy/Desktop/Code/Algoritms/task5/text2.txt";

ifstream Reader(Path1);

Queue<string>\* WordsQueue = new Queue<string>("", 100);

string Word;

while (Reader >> Word)

{

WordsQueue->PushBack(Word);

}

WordsQueue->PopFront();

ofstream Writer(Path2);

count = WordsQueue->Count;

for (size\_t i = 0; i < count; i++)

{

Writer << WordsQueue->PopFront() + "\n";

}

//предикаты 1

Queue<int>\* q3 = new Queue<int>(1, 5);

q3->PushBack(3);

q3->PushBack(5);

q3->PushBack(7);

q3->PushBack(9);

cout << q3->IsContains(1) << endl;

q3->PopFront();

cout << q3->IsContains(1) << endl;

cout << q3->IsContains(3) << endl;

q3->PopFront();

cout << q3->IsContains(3) << endl;

cout << q3->IsContains(7) << endl;

return 0;

}

**Тест программы**

Тест программы представлен на рисунке 1. Содержимое файлов, с которыми работает программа, представлено на рисунках 2-3.

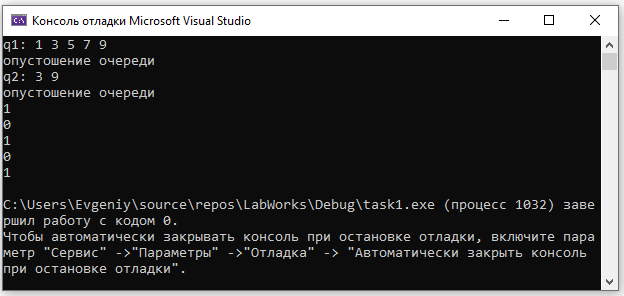


Рисунок 1 - Тест программы 1

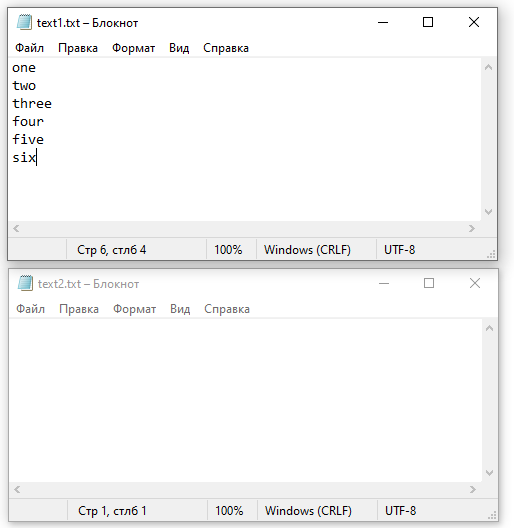


Рисунок 2 – Содержимое файлов до запуска программы

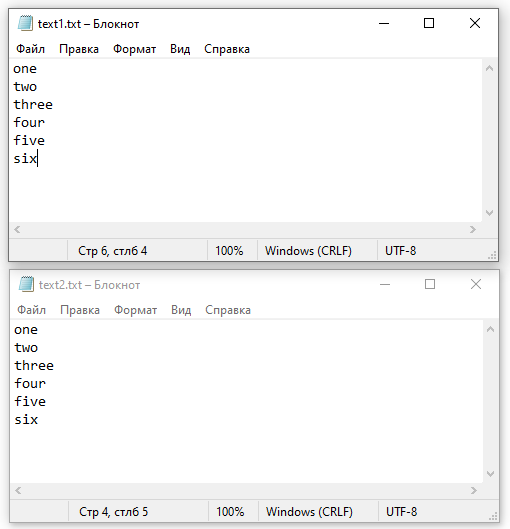


Рисунок 3 - Содержимое файлов после запуска программы