МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Кафедра информатики и систем управления

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Лабораторная работа №1\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование темы проекта или работы)

ОТЧЁТ

По лабораторной работе

по дисциплине

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Алгоритмы и структуры данных\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование дисциплины)

РУКОВОДИТЕЛЬ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (фамилия, и.,о.)

СТУДЕНТ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (фамилия, и.,о.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(шифр группы)

Работа защищена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород 2022

**Задание:**

Используя очередь, отредактировать текст, оставляя один пробел в каждой серии

пробелов.

**Список функций и структур данных с описанием:**

Класс очередь - структура данных (как было сказано выше), которая построена по принципу **LILO** (last in — last out: последним пришел — последним вышел). В очереди, если добавить элемент, который вошел самый первый, то он выйдет тоже самым первым.

Out- функция вывода всех элементов очереди

Queue::push - Функция добавления элемента в "хвост" очереди

Queue::pop - Функция удаления элемента из "головы" очереди

Queue::size - Функция определения длины очереди

Queue::front - Функция возврата первого элемента очереди

**Программный код:**

#include <iostream>

#include <iomanip>

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// Класс очередь

class Queue{

static const int SIZE=10010; // максимальный размер очереди

char\* queue; // хранилище элементов очереди

int first,last; // номер первого элемента очереди, номер последнего элемента очереди

public:

Queue(); // базовый конструктор

void push(char num); // функция добавления элементов

friend void out(Queue q); // функция вывода

int size(); // функция получения размера

void pop(); // функция удаления первого элемета

char front(); // функция чтения первого элемента

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

Queue::Queue()

{

queue = new char[SIZE];

first = last = 0 ;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// Функция вывода всех элементов очереди

void out(Queue q){

for (int i=q.first+1;i<q.last+1;i++){

std::cout<<q.queue[i];

}

}

// Функция добавления элемента в "хвост" очереди

void Queue::push ( char num )

{

// проверка полноты очереди

if ( last+1==first || ( last+ 1 ==SIZE && !first )) {

std::cout << "Очередь полна\n" ;

return ;

}

last++;

if ( last==SIZE ){

last = 0 ;

}

queue [ last ] = num;

}

// Функция удаления элемента из "головы" очереди

void Queue::pop()

{

// проверка на непустую очередь

if ( first == last ) {

std::cout << "Очередь пуста\n" ;

return ;

}

first++;

// проверка перезаписи очереди

if ( first == SIZE ){

first = 0 ;

}

}

// Функция определения длины очереди

int Queue::size()

{ int s = 0;

// цикл продсчёта количества элементов

for (int i=first ; i<last ; i++)

{

s++;

}

return s;

}

// Функция возврата первого элемента очереди

char Queue::front()

{

return queue[first+1];

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// Основная функция

int main(){

char logo[] =

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n"

"\* Nizhny Novgorog State Technical University \*\n"

"\* Study work number 1. Task number 1. \*\n"

"\* Performed student 21-IVT-2 Kitov A. A. \*\n"

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

setlocale(LC\_ALL, "Russian");

std::cout<< logo <<std::endl;

// Создание объекта типа очередь

Queue queue1;

char buf[10000];

for (int i = 0; i < 10000; i++) { /////////////////////////////////////

buf[i] = '0'; // Блок зануления пустых элементов //

} /////////////////////////////////////

std::cout<<"Введите текст:"; ///////////////////////

std::cin.getline(buf,10000); // Блок ввода текста //

std::cout<<std::endl; ///////////////////////

int k=0, len=0, ex=0;

// Цикл проходки по всем элементам введённого теста

for (int i=0 ; i<10000 ; i++){

ex=(int)buf[i];

if(buf[i]==' ' and k==0 and buf[i]!='0'){ ///////////////////////////////////

queue1.push(buf[i]); // //

k = k + 1; // //

} // Блок ввода элементов в строку //

if(ex>64 and ex<123 and buf[i]!='0'){ // с удалением лишних пробелов //

queue1.push(buf[i]); // //

k = 0; // //

} ///////////////////////////////////

}

// Вывод итогового преобразованного текста

std::cout<< std::endl<< "Итог:" <<std::endl;

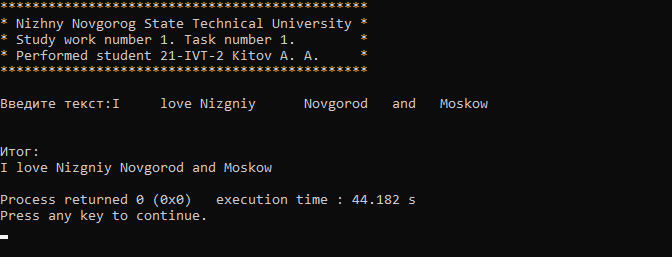
out(queue1);

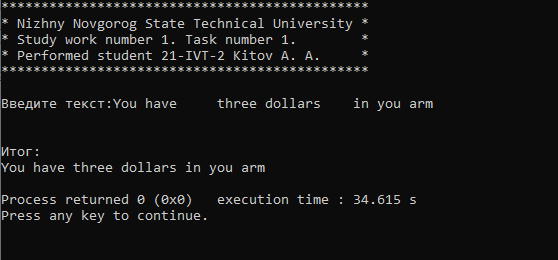
std::cout<<std::endl;

return 0;

}

**Результаты работы программы:**





P.s. проверка на ввод отсутствует, потому что текст может состоять из любых символов.