**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ВЫСШАЯ ШКОЛА ПЕЧАТИ И МЕДИАИНДУСТРИИ**

***Институт Принтмедиа и информационных технологий***

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ РАБОТА № 1**

**Дисциплина: Основы современных алгоритмов**

**Тема: "Стеки и очереди"**

**Цель:** **Получить практические навыки в использовании стеков и очередей в языке С**

**Выполнил(а): студент группы 201-726**

**Забалуев Владислав**

(Фамилия И.О.)

**Дата, подпись** 11 сентября 2021  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Проверил: \_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_***

(Фамилия И.О., степень, звание) **(Оценка)**

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва**

**2021**

**Задача:**

Необходимо разработать программу, реализующую работу со стеком очередью.

Программа должна содержать функции вставки, извлечения, контроля на переполнение, контроля на отсутствие элементов (пустой стек или очередь), отображения на экране значения извлекаемого элемента.

#include <iostream>

class Stack

{

private:

int N = 10;

int\* arr;

int top; //переменная для вершины стэка

public:

Stack()

{

top = -1;

arr = new int[N];

}

void Push(int chislo)

{

if (top == N - 1) //если значение вершины равно размеру массива, то

{

printf("Stack Overflow \n");

}

else //иначе добавляем в стек

{

top++;

arr[top] = chislo;

}

}

void Pop()

{

if (top == -1) //если элементов в стеке нет, то

{

printf("Stack null \n");

}

else

{

printf("%d \n", arr[top]); //иначе выводим элемент

top--;

}

}

};

class List

{

private:

int N = 3;

int\* arr;

int top; //переменные головы и хвоста

int bot;

public:

List()

{

top = 0;

bot = 0;

arr = new int[N];

}

void Add(int chislo)

{

if (top == (bot + 1) % N) //если в очереди есть место, то

{

printf("List overflow \n");

}

else

{

arr[bot] = chislo; //добавляем число

bot = (bot + 1) % N;

}

}

void Remove()

{

if (top != bot) //если в очереди есть элементы, то выводим их

{

printf("%d \n", arr[top]);

top = (top + 1) % N;

}

else

{

printf("List null \n");

}

}

};

int main()

{

Stack stack;

stack.Pop();

stack.Push(15);

stack.Push(1);

stack.Push(5);

stack.Pop();

stack.Pop();

stack.Pop();

printf("\nNow List \n \n");

List list;

list.Remove();

list.Add(63);

list.Add(3);

list.Add(3);

list.Remove();

list.Remove();

list.Remove();

}

