Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

Создать класс для объекта стек. Стек хранит целые числа. Имеет характеристики: наименование (строка, не более 10 символов) и размер (целое). Размер стека больше или равно 1. Функционал стека:

- добавить элемент и вернуть признак успеха (логическое);
- извлечь элемент (НЕ вывести!) и вернуть признак успеха (логическое);
- получить имя стека (строка);
- получить размер стека (целое);
- получить текущее количество элементов в стеке (целое).

В классе определить параметризированный конструктор, которому передается имя стека и размер. При переполнении стека очередной элемент не добавлять и определяется соответствующий признак успеха.

В основной программе реализовать алгоритм:

1. Ввести имя и размер для первого стека.
2. Создать объект первого стека.
3. Ввести имя и размер для второго стека.
4. Создать объект второго стека.
5. В цикле:

5.1. Считывать очередное значение элемента.

5.2. Добавлять элемент в первый стек, при переполнении завершить цикл.

5.3. Добавлять элемент во второй стек, при переполнении завершить цикл.

6. Построчно вывести содержимое стеков.

1.1 Описание входных данных

Первая строка:

«имя стека 1» «размер стека»

Вторая строка:

«имя стека 2» «размер стека»

Третья строка:

Последовательность целых чисел, разделенных пробелами, в количестве не менее чем размер одного из стеков + 1.

1.2 Описание выходных данных

Первая строка:

«имя стека 1» «размер»

Вторая строка:

«имя стека 2» «размер»

Третья строка:

«имя стека 1»«имя стека 2»

Каждое имя стека в третьей строке занимает поле длины 15 позиции и прижата к левому краю.

Четвертая строка и далее построчно, вывести все элементы стеков:

«значение элемента стека 1»«значение элемента стека 2»

Вывод значений элементов стеков производиться последовательным извлечением.

Каждое значение занимает поле из 15 позиции и прижата к правому краю.

2 МЕТОД РЕШЕНИЯ

Для решения задачи понадобится:

- библиотека iomanip (функция setw)
- библиоткека string
- используется оператор функци new
- условный оператор if
- объекты двух стеков
- оператор цикла while
- оператор break
- объект потока ввода cin и объект потока вывода cout

Класс Stack

Поля:

доступные элементы с типом string name, int size

скрытые элементы с типом int count, int *array (указатель массива)

Методы:

Открытые:

Stack(string name, int size) - конструктор, принимающий имя стека и размер

bool input(int element) - добавляет элемент в стек

bool output(int &where) - извлекает элемент стека

string get_name() - получает имя стека

int get_size() - получает размер стека

int out_count() - получает текущее количество элементов в стеке

3 ОПИСАНИЕ АЛГОРИТМОВ

Согласно этапам разработки, после определения необходимого инструментария в разделе «Метод», составляются подробные описания алгоритмов для методов классов и функций.

3.1 Алгоритм функции main

Функционал: главный метод программы.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм функции представлен в таблице 1.

Таблица 1 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		ввод имени и размера первого стека	2
2		создание объекта первого стека	3
3		ввод имени и размера второго стека	4
4		создание объекта второго стека	5
5		считывание значение элемента	6
6	при переполнении завершить цикл	добавление элемента в первый стек и во второй стек	5
			7
7		вывод имени первого стека и размер	8
8		вывод имени второго стека и размер	9
9		вывод имени первого стека и второго стека по левому краю	10
10		вывод построчно содержимого стеков	∅

3.2 Алгоритм конструктора класса Stack

Функционал: string name, int size.

Параметры: .

Алгоритм конструктора представлен в таблице 2.

Таблица 2 – Алгоритм конструктора класса Stack

№	Предикат	Действия	№ перехода
1		присваивание закрытому свойству name значение аргумента name	2
2		присваивание закрытому свойству size значение аргумента size	3
3		создание массива array	∅

3.3 Алгоритм метода input класса Stack

Функционал: добавление элемента в стек и возвращение признака успеха.

Параметры: int element.

Возвращаемое значение: bool.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода input класса Stack

№	Предикат	Действия	№ перехода
1	индекс + 1 больше размера стека	возвращение false	∅
		присвоение элемента массива array по индексу count значение element	2
2		увеличение счетчика на один	3
3		возвращение true	∅

3.4 Алгоритм метода output класса Stack

Функционал: извлечение элемента и возвращение признака успеха.

Параметры: int &where.

Возвращаемое значение: bool.

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода output класса Stack

№	Предикат	Действия	№ перехода
1	индекс - 1 < 0	возвращение false	∅
		уменьшение счетчика на один	2
2		присвоение where значение элемента массива array по индексу count	3
3		возвращение true	∅

3.5 Алгоритм метода get_name класса Stack

Функционал: возвращение имя стека.

Параметры: нет.

Возвращаемое значение: string.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода get_name класса Stack

№	Предикат	Действия	№ перехода
1		возвращение имя стека	∅

3.6 Алгоритм метода get_size класса Stack

Функционал: возвращение размера стека.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 6.

Таблица 6 – Алгоритм метода get_size класса Stack

№	Предикат	Действия	№ перехода
1		возвращение размера стека	∅

3.7 Алгоритм метода out_count класса Stack

Функционал: получение текущего количества элементво в стеке.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода out_count класса Stack

№	Предикат	Действия	№ перехода
1		возвращение текущего количества элементов в стеке	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-4.

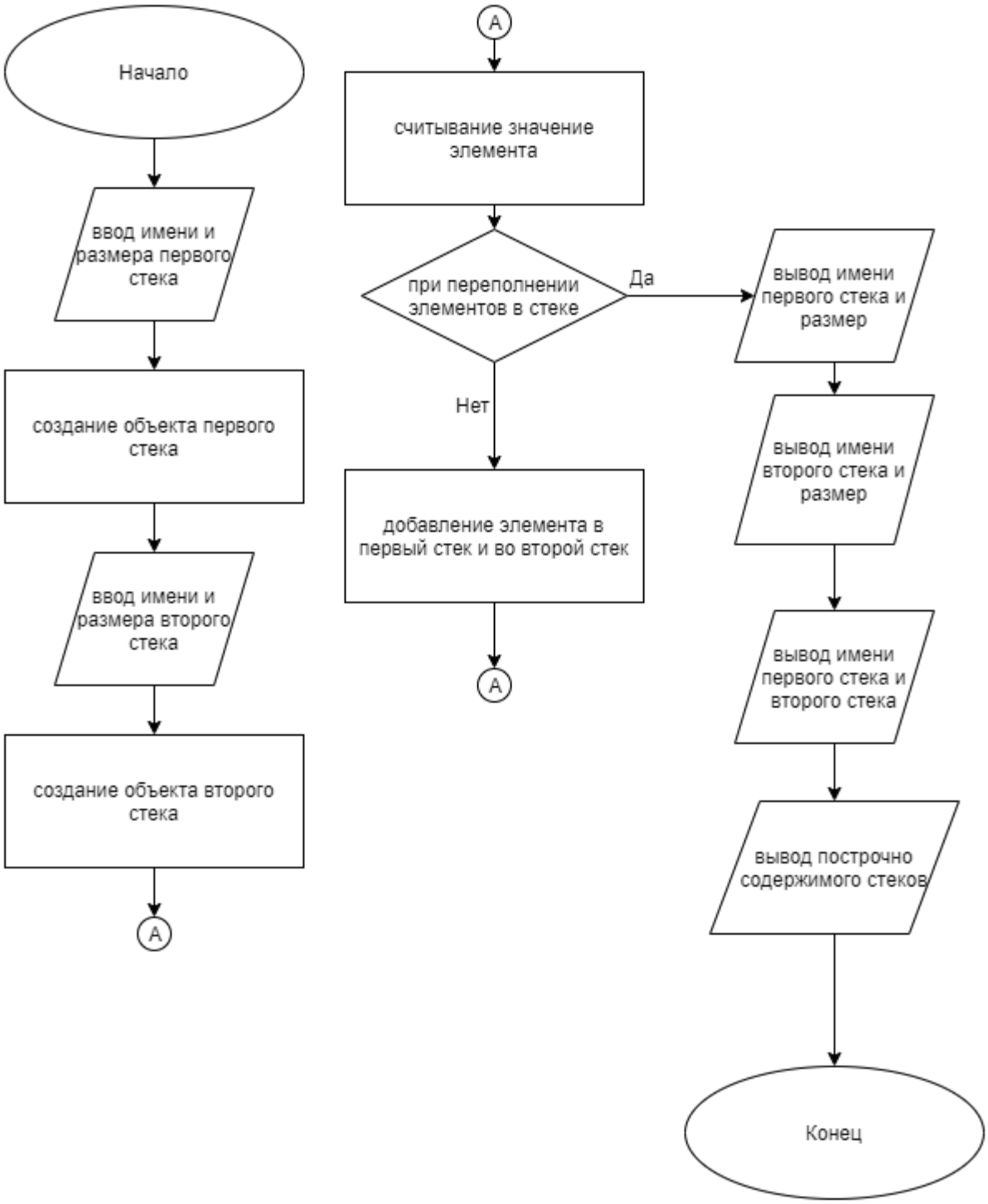

Рисунок 1 – Блок-схема алгоритма

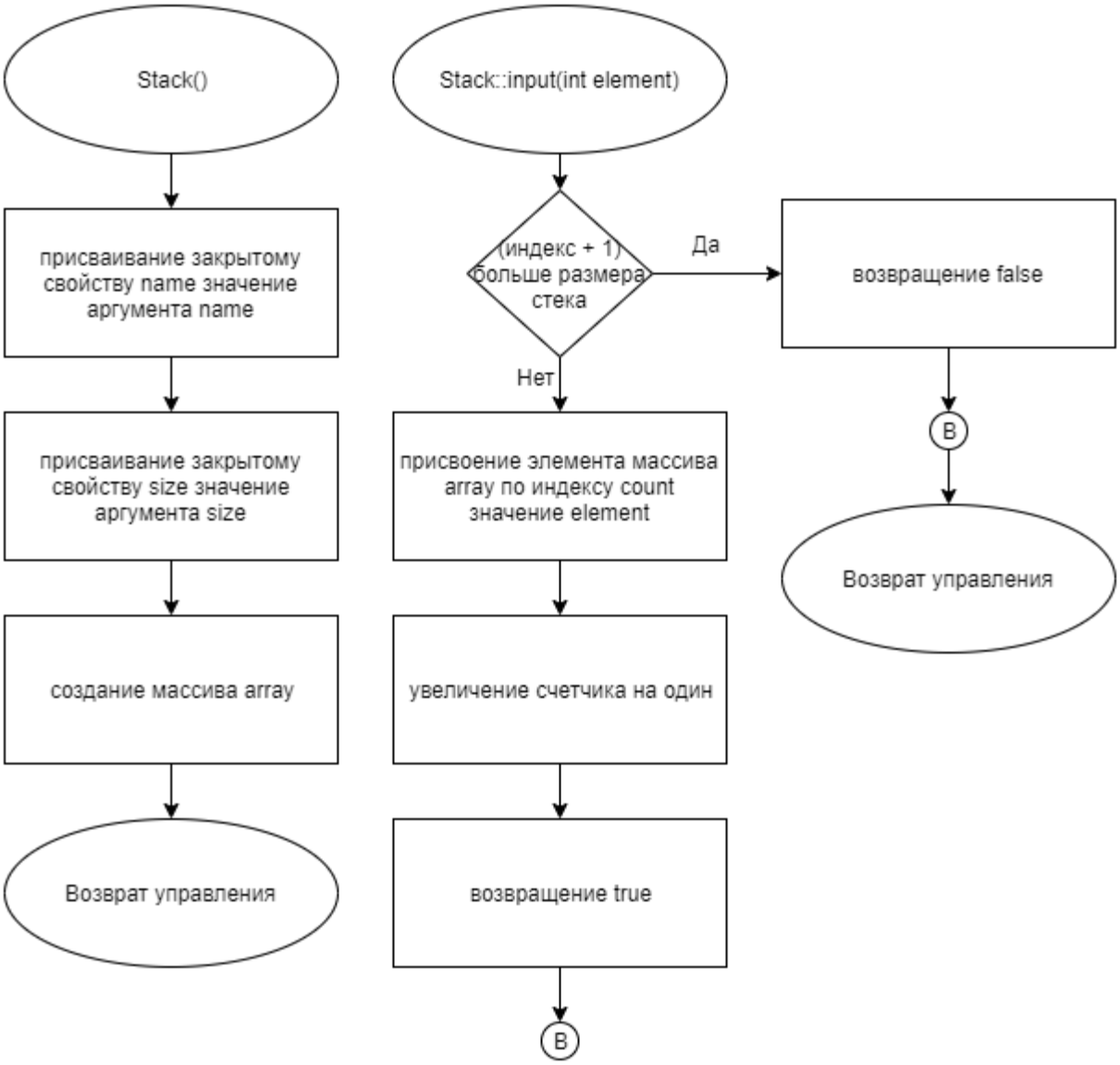

Рисунок 2 – Блок-схема алгоритма

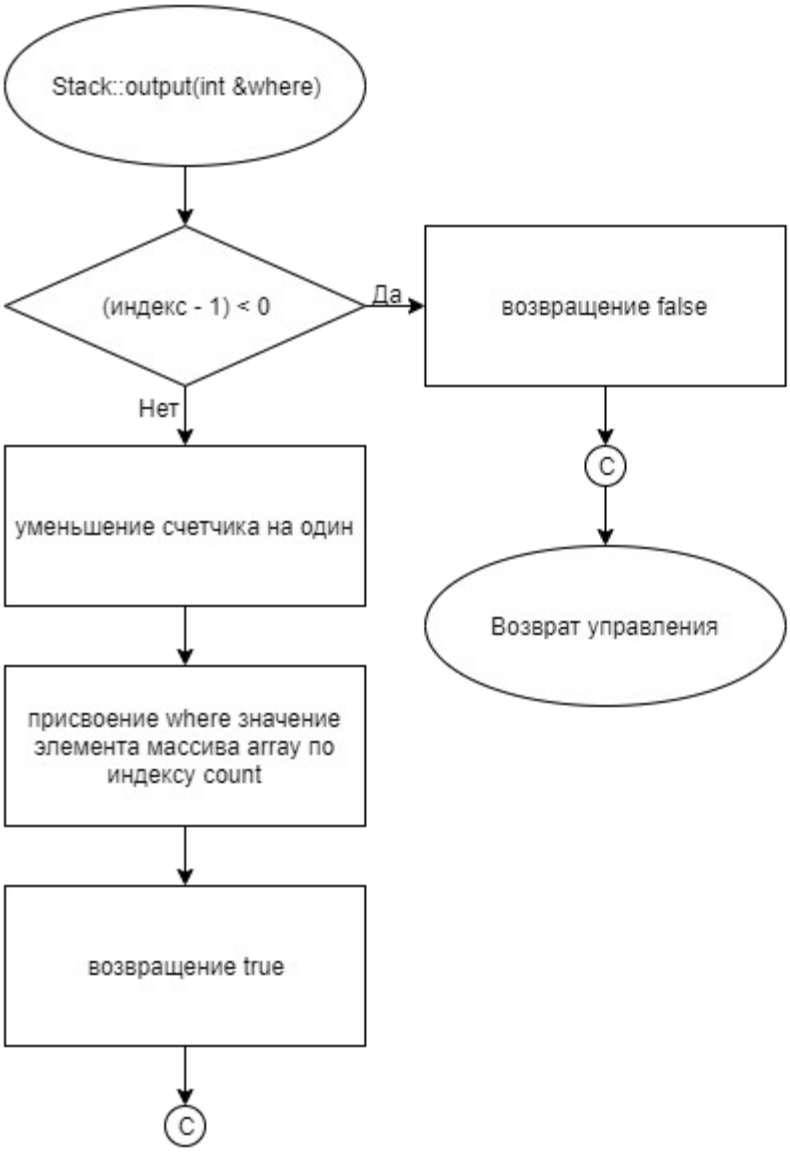

Рисунок 3 – Блок-схема алгоритма

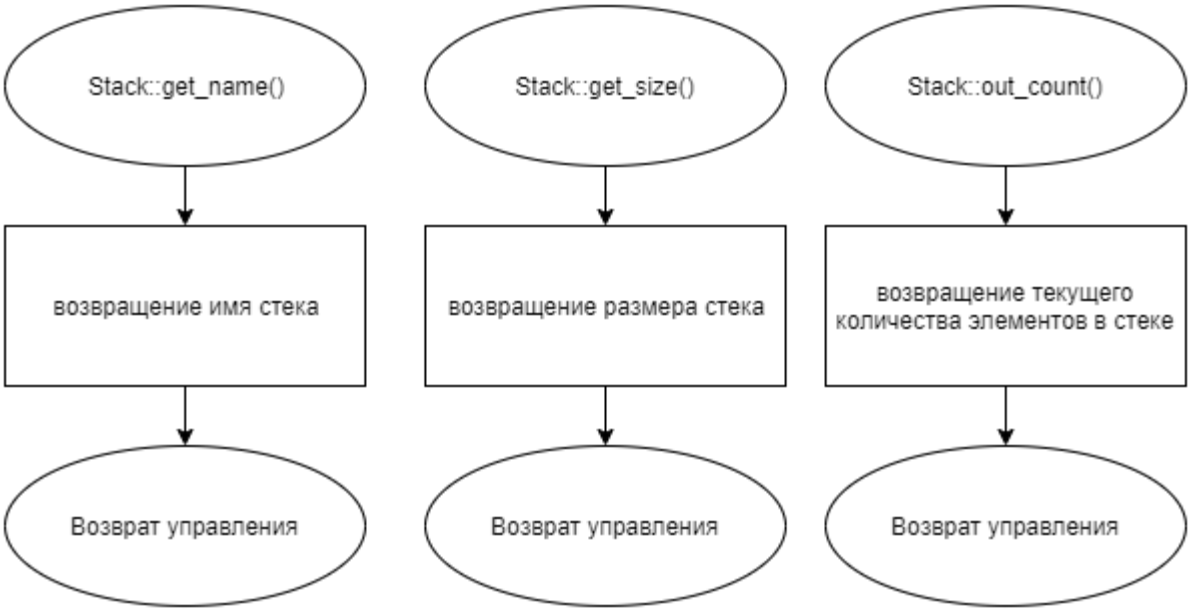

Рисунок 4 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл main.cpp

Листинг 1 – main.cpp

```
#include <stdlib.h>
#include <stdio.h>
#include <iostream>
#include <string>
#include <iomanip>
#include "Stack.h"

using namespace std;

int main()
{
	string name;
	int size;
	cin >> name >> size;

	Stack stack1 = Stack(name, size);

	cin >> name >> size;

	Stack stack2 = Stack(name, size);

	while (true)
	{
		int element;
		cin >> element;
		if (stack1.input(element) == false || stack2.input(element) == false)
		{
			break;
		}
	}

	cout << stack1.get_name() << " " << stack1.get_size() << endl;
	cout << stack2.get_name() << " " << stack2.get_size() << endl;

	cout << left << setw(15) << stack1.get_name();
	cout << left << setw(15) << stack2.get_name();

	int element1, element2;
	stack1.output(element1);

	while (true)
```

```
	{
		cout << endl;
		if (stack2.output(element2))
		{
			cout << setw(15) << right << element1;
		}
		else
		{
			cout << setw(15) << right << element1;
			break;
		}
		if (stack1.output(element1))
		{
			cout << setw(15) << right << element2;
		}
		else
		{
			cout << setw(15) << right << element2;
			break;
		}
	}
	return(0);
}
```

5.2 Файл Stack.cpp

Листинг 2 – Stack.cpp

```
#include <iostream>
#include <string>
#include <iomanip>
#include "Stack.h"

using namespace std;

Stack::Stack(string name, int size)
{
	this->name = name;
	this->size = size;
	array = new int[size];
}

bool Stack::input(int element)
{
	if (count + 1 > size)
	{
		return false;
	}
	else
	{
		array[count] = element;
		count++;
		return true;
```

```
	}
}

bool Stack::output(int& where)
{
	if (count - 1 < 0)
	{
		return false;
	}
	else
	{
		count--;
		where = array[count];
		return true;
	}
}

string Stack::get_name()
{
	return name;
}

int Stack::get_size()
{
	return size;
}

int Stack::out_count()
{
	return count;
}
```

5.3 Файл Stack.h

Листинг 3 – Stack.h

```
#ifndef __STACK_H
#define __STACK_H

using namespace std;

class Stack
{
private:
	int *array;
	int count = 0;
public:
	string name;
	int size;
	Stack(string name, int size);
	bool input(int element);
	bool output(int& where);
	string get_name();
```

```
      int get_size();
      int out_count();
};

#endif
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 8.

Таблица 8 – Результат тестирования программы

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
S1 3 S2 2 1 2 3 4 5 6 7 8 9 0	S1 3 S2 2 S1 S2 3 2 2 1 1	S1 3 S2 2 S1 S2 3 2 2 1 1

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Васильев А.Н. Объектно-ориентированное программирование на C++. Издательство: Наука и Техника. Санкт-Петербург, 2016г. 543 стр.

2. Шилдт Г. C++: базовый курс. 3-е изд. Пер. с англ.. — М.: Вильямс, 2017. — 624 с.

3. Методическое пособие для проведения практических заданий, контрольных и курсовых работ по дисциплине «Объектно-ориентированное программирование» [Электронный ресурс] – URL: https://mirea.aco-avrora.ru/student/files/methodichescoe_posobie_dlya_laboratorny h_rabot_3.pdf (дата обращения 05.05.2021).

4. Приложение к методическому пособию студента по выполнению заданий в рамках курса «Объектно-ориентированное программирование» [Электронный ресурс]. URL: https://mirea.aco-avrora.ru/student/files/Prilozheniye_k_methodichke.pdf (дата обращения 05.05.2021).

5. Видео лекции по курсу «Объектно-ориентированное программирование» [Электронный ресурс]. АСО «Аврора».

6. Антик М.И. Дискретная математика [Электронный ресурс]: Учебное пособие /Антик М.И., Казанцева Л.В. — М.: МИРЭА — Российский технологический университет, 2018 — 1 электрон. опт. диск (CD-ROM).