МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

« МИРЭА Российский технологический университет»

РТУ МИРЭА

Институт Информационных технологий

Кафедра Вычислительной техники

УЧЕБНОЕ ЗАДАНИЕ

по дисциплине

« Объектно-ориентированное программирование»

Наименование задачи:

« Задание 4_2_1 »

С тудент группы	ИКБО-13-21	Дамарад Д.В.
Руководитель практики	Ассистент	Асадова Ю.С.
Работа представлена	«___»_____ 2022 г.	_____
		(подпись студента)
Оценка		_____
		(подпись руководителя)

Москва 2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Постановка задачи

Множественное наследование

Даны 8 классов, которые нумеруются от 1 до 8. Классы 2, 3, 4 и 5 наследованы от первого класса. Шестой класс от второго и третьего. Седьмой от четвертого и пятого. Восьмой от шестого и седьмого.
У каждого класса есть параметризированный конструктор с одним параметром строкового типа и закрытое свойство строкового типа для наименования объекта класса.
Значение данного свойства определяется в параметризированном конструкторе согласно шаблону:

«значение строкового параметра»_«номер класса»

В основной функции реализовать алгоритм:

1. Объявить один указатель на объект класса x (где: x - номер класса, его надо определить).
2. Объявить переменную строкового типа.
3. Ввести значение строковой переменной. Вводимое значение является идентификатором.
4. Создать объект класса 8 посредством параметризированного конструктора, передав в качестве аргумента строковую переменную.
5. Адрес созданного объекта присвоить указателю на объект класса x.
6. Используя только указатель на объект класса x вывести имена всех объектов в составе объекта класса 8 и имя самого объекта класса 8. Вывод выполнить построчно, упорядочивая согласно возрастанию номеров класса. Вывод реализовать в основной функции.

Наследственность реализовать так, чтобы всего объектов было 10.

Описание входных данных

Первая строка:

«идентификатор»

Пример ввода

Ident

Описание выходных данных

Построчно (десять строк):

«идентификатор»_«номер класса»

Пример вывода:

Ident_1

Ident_1

Ident_1

Ident_2

Ident_3

Ident_4

Ident_5

Ident_6

Ident_7

Ident_8

Метод решения

Для решения поставленной задачи используются:

- Потоки стандартного ввода и вывода cin и cout соответственно, используются для ввода и вывода на экран.
- Объект класса Class8 для создания всех отсальных объектов.
- Указатель ptr на тип Class8 для вывода имен объектов всех созданных классов.

Класс Class1:

- Поля:
    - Поле, хранящее имя объекта:
        - Наименование - name;
        - Тип - строковый;
        - Модификатор доступа - private.
- Методы:
    - Метод - GetName:
        - Функционал - геттер поля name.
    - Конструктор Class1:
        - Функционал - параметризированный конструктор.

Класс Class2:

- Поля:
    - Поле, хранящее имя объекта:
        - Наименование - name;
        - Тип - строковый;
        - Модификатор доступа - private.
- Методы:

        - Конструктор Class2:
            - Функционал - параметризированный конструктор.

Класс Class3:

- Поля:
    - Поле, хранящее имя объекта:
        - Наименование - name;
        - Тип - строковый;
        - Модификатор доступа - private.
- Методы:
    - Конструктор Class3:
        - Функционал - параметризированный конструктор.

Класс Class4:

- Поля:
    - Поле, хранящее имя объекта:
        - Наименование - name;
        - Тип - строковый;
        - Модификатор доступа - private.
- Методы:
    - Конструктор Class4:
        - Функционал - параметризированный конструктор.

Класс Class5:

- Поля:
    - Поле, хранящее имя объекта:

            - Наименование - name;
            - Тип - строковый;
            - Модификатор доступа - private.
    - Методы:
        - Конструктор Class5:
            - Функционал - параметризированный конструктор.

Класс Class6:

- Поля:
    - Поле, хранящее имя объекта:
        - Наименование - name;
        - Тип - строковый;
        - Модификатор доступа - private.
- Методы:
    - Конструктор Class6:
        - Функционал - параметризированный конструктор.

Класс Class7:

- Поля:
    - Поле, хранящее имя объекта:
        - Наименование - name;
        - Тип - строковый;
        - Модификатор доступа - private.
- Методы:
    - Конструктор Class7:
        - Функционал - параметризированный конструктор.

Класс Class8:

- Поля:
    - Поле, хранящее имя объекта:
        - Наименование - name;
        - Тип - строковый;
        - Модификатор доступа - private.
- Методы:
    - Конструктор Class8:
        - Функционал - параметризированный конструктор.

№	Имя класса	Классы-наследники	Модификаторы доступы при наследовании	Описание	Номер	Комментарий
1	Class1	Class2 Class3 Class4 Class5	public	Базовый класс в иерархии классов	2 3 4 5	
2	Class2	Class6	public	Класс объектов, подчиненных классу Class1	6	
3	Class3	Class6	public	Класс объектов, подчиненных классу Class1	6	
4	Class4	Class7	public	Класс объектов, подчинен	7	

				ных классу Class1		
5	Class5	Class7	public	Класс объектов, подчиненных классу Class1	7	
6	Class6	Class8	public	Класс объектов, подчиненных классу Class2 и Class3	8	
7	Class7	Class8	public	Класс объектов, подчиненных классу Class4 и Class5	8	
8	Class8			Класс объектов, подчиненных классу Class6 и Class7		

Описание алгоритма

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

Функция: main

Функционал: Основной алгоритм программы

Параметры: Отсутсвуют

Возвращаемое значение: Целочисленное значение - код возврата

Алгоритм функции представлен в таблице 2.

Таблица 2. Алгоритм функции main

№	Предикат	Действия	№ перехода	Комментарий
1		Объявление строковой переменной name	2	
2		Считывание с клавиатуры значения переменной name	3	
3		Создание объекта obj8 класса Class8 с параметром name	4	
4		Объявление указателя ptr на класс Class8 с присвоением ему ссылки на объект obj8	5	
5		Вывод на экран результатов вызова метода GetName по указателю ptr всех созданных элементов	∅	

Класс объекта: Class1

Модификатор доступа: public

Метод: GetName

Функционал: Геттер поля name

Параметры: Отсутсвуют

Возвращаемое значение: string

Алгоритм метода представлен в таблице 3.

Таблица 3. Алгоритм метода GetName класса Class1

№	Предикат	Действия	№ перехода	Комментарий
1		Возврат значения поля name	∅	

Конструктор класса: Class1

Модификатор доступа: public

Функционал: Параметризированный конструктор

Параметры: Строковая переменная - name

Алгоритм конструктора представлен в таблице 4.

Таблица 4. Алгоритм конструктора класса Class1

№	Предикат	Действия	№ перехода	Комментарий
1		Присвоение полю name параметра name + "_1"	∅	

Конструктор класса: Class2

Модификатор доступа: public

Функционал: Параметризированный конструктор

Параметры: Строковая переменная - name

Алгоритм конструктора представлен в таблице 5.

Таблица 5. Алгоритм конструктора класса Class2

№	Предикат	Действия	№ перехода	Комментарий
1		Присвоение полю name параметра name + "_2"	∅	

Конструктор класса: Class3

Модификатор доступа: public

Функционал: Параметризированный конструктор

Параметры: Строковая переменная - name

Алгоритм конструктора представлен в таблице 6.

Таблица 6. Алгоритм конструктора класса Class3

№	Предикат	Действия	№ перехода	Комментарий
1		Присвоение полю name параметра name + "_3"	∅	

Конструктор класса: Class4

Модификатор доступа: public

Функционал: Параметризированный конструктор

Параметры: Строковая переменная - name

Алгоритм конструктора представлен в таблице 7.

Таблица 7. Алгоритм конструктора класса Class4

№	Предикат	Действия	№ перехода	Комментарий

1		Присвоение полю name параметра name + "_4"	∅	

Конструктор класса: Class5

Модификатор доступа: public

Функционал: Параметризированный конструктор

Параметры: Строковая переменная - name

Алгоритм конструктора представлен в таблице 8.

Таблица 8. Алгоритм конструктора класса Class5

№	Предикат	Действия	№ перехода	Комментарий
1		Присвоение полю name параметра name + "_5"	∅	

Конструктор класса: Class6

Модификатор доступа: public

Функционал: Параметризированный конструктор

Параметры: Строковая переменная - name

Алгоритм конструктора представлен в таблице 9.

Таблица 9. Алгоритм конструктора класса Class6

№	Предикат	Действия	№ перехода	Комментарий
1		Присвоение полю name параметра name + "_6"	∅	

Конструктор класса: Class7

Модификатор доступа: public

Функционал: Параметризированный конструктор

Параметры: Строковая переменная - name

Алгоритм конструктора представлен в таблице 10.

Таблица 10. Алгоритм конструктора класса Class7

№	Предикат	Действия	№ перехода	Комментарий
1		Присвоение полю name параметра name + "_7"	∅	

Конструктор класса: Class8

Модификатор доступа: public

Функционал: Параметризированный конструктор

Параметры: Строковая переменная - name

Алгоритм конструктора представлен в таблице 11.

Таблица 11. Алгоритм конструктора класса Class8

№	Предикат	Действия	№ перехода	Комментарий
1		Присвоение полю name параметра name + "_8"	∅	

Блок-схема алгоритма

Представим описание алгоритмов в графическом виде на рисунках ниже.

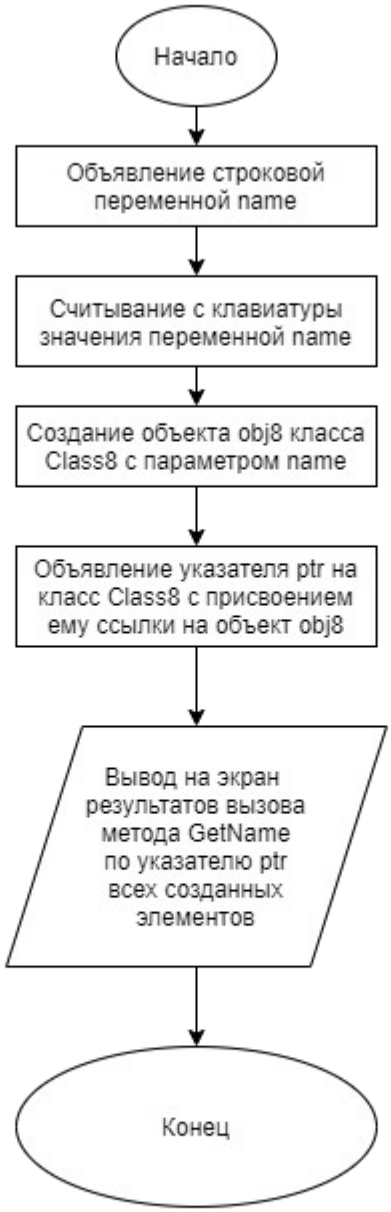

Рис. 1. Блок-схема алгоритма.

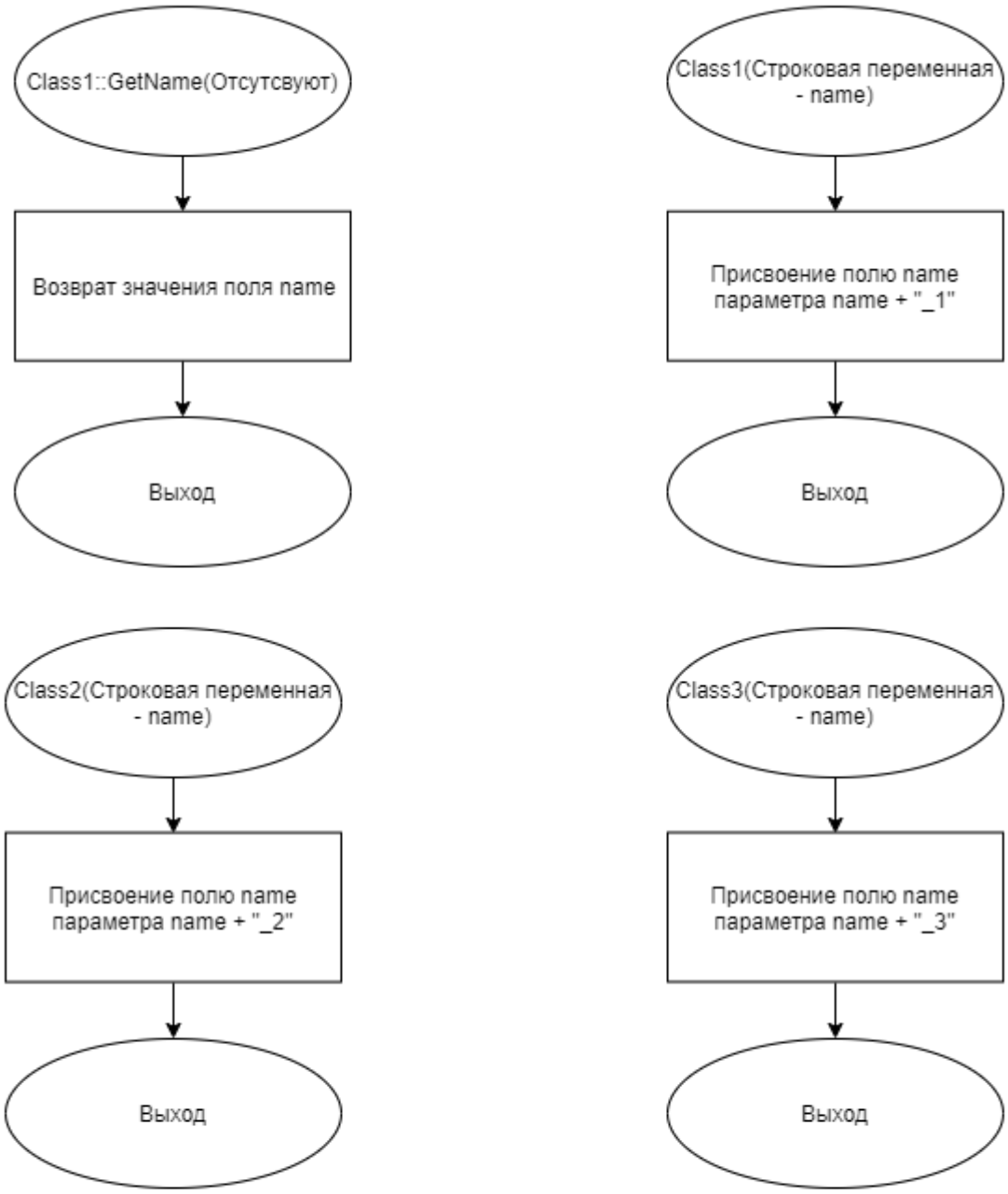

Рис. 2. Блок-схема алгоритма.

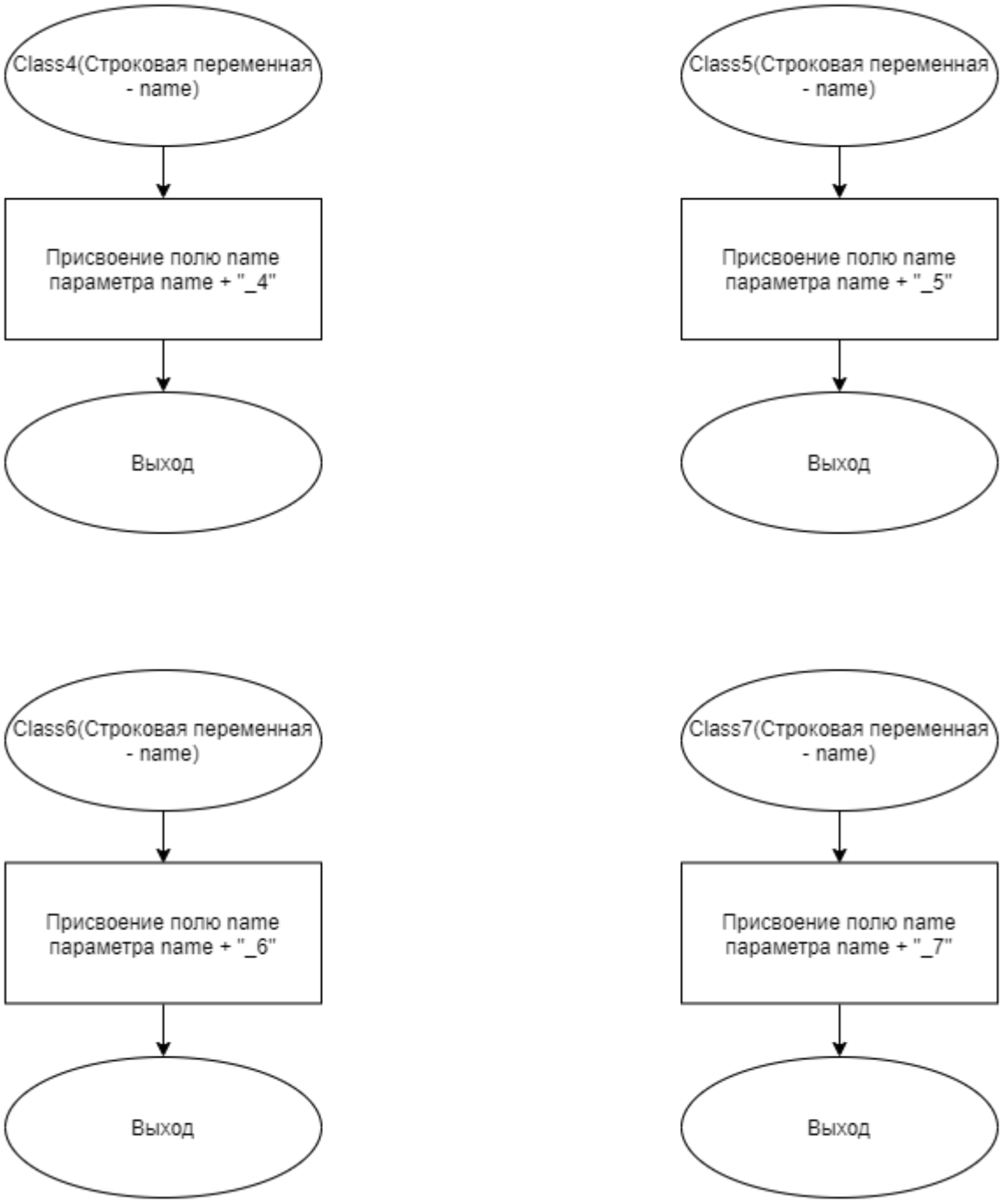

Рис. 3. Блок-схема алгоритма.

Рис. 4. Блок-схема алгоритма.

Код программы

Программная реализация алгоритмов для решения задачи представлена ниже.

Файл Class1.cpp

```
#include "Class1.h"
#include <iostream>
Class1::Class1(string name){
        this->name=name + "_1";
}
string Class1::GetName(){
        return this->name;
}
```

Файл Class1.h

```
#ifndef _CLASS1_H
#define _CLASS1_H
#include <string>
using namespace std;
class Class1{
private:
        string name;
public:
        Class1(string name);
        string GetName();
};
#endif
```

Файл Class2.cpp

```
#include "Class2.h"
#include <iostream>
Class2::Class2(string name):Class1(name){
        this->name=name + "_2";
}
```

Файл Class2.h

```
#ifndef _CLASS2_H
```

```
#define _CLASS2_H
#include "Class1.h"
class Class2:public virtual Class1{
private:
        string name;
public:
        Class2(string name);
        string GetName() { return name; }
};
#endif
```

Файл Class3.cpp

```
#include "Class3.h"
#include <iostream>
Class3::Class3(string name):Class1(name){
        this->name=name + "_3";
}
```

Файл Class3.h

```
#ifndef _CLASS3_H
#define _CLASS3_H
#include "Class1.h"
class Class3:public virtual Class1{
private:
        string name;
public:
        Class3(string name);
        string GetName() { return name; }
};
#endif
```

Файл Class4.cpp

```
#include "Class4.h"
#include <iostream>
Class4::Class4(string name):Class1(name){
        this->name=name + "_4";
}
```

Файл Class4.h

```
#ifndef _CLASS4_H
#define _CLASS4_H
#include "Class1.h"
class Class4:public Class1{
private:
        string name;
public:
        Class4(string name);
        string GetName() { return name; }
};
#endif
```

Файл Class5.cpp

```
#include "Class5.h"
#include <iostream>
Class5::Class5(string name):Class1(name){
        this->name=name + "_5";
}
```

Файл Class5.h

```
#ifndef _CLASS5_H
#define _CLASS5_H
#include "Class1.h"
class Class5:public Class1{
private:
        string name;
public:
        Class5(string name);
        string GetName() { return name; }
};
#endif
```

Файл Class6.cpp

```
#include "Class6.h"
#include <iostream>
Class6::Class6(string name):Class2(name),Class3(name),Class1(name){
        this->name=name + "_6";
}
```

Файл Class6.h

```
#ifndef _CLASS6_H
#define _CLASS6_H
#include "Class2.h"
#include "Class3.h"
class Class6:public Class2, public Class3{
private:
        string name;
public:
        Class6(string name);
        string GetName() { return name; }
};
#endif
```

Файл Class7.cpp

```
#include "Class7.h"
#include <iostream>
Class7::Class7(string name):Class4(name), Class5(name){
        this->name=name + "_7";
}
```

Файл Class7.h

```
#ifndef _CLASS7_H
#define _CLASS7_H
#include "Class4.h"
#include "Class5.h"
class Class7:public Class4, public Class5{
private:
        string name;
public:
        Class7(string name);
        string GetName() { return name; }
};
#endif
```

Файл Class8.cpp

```
#include "Class8.h"
#include <iostream>
Class8::Class8(string name):Class6(name),Class7(name),Class6::Class1(name){
        this->name=name + "_8";
}
```

Файл Class8.h

```
#ifndef _CLASS8_H
#define _CLASS8_H
#include "Class6.h"
#include "Class7.h"
class Class8:public Class6, public Class7{
private:
        string name;
public:
        Class8(string name);
        string GetName() { return name; }
};
#endif
```

Файл main.cpp

```
#include <iostream>
#include <string>
#include "Class8.h"
int main()
{
        string name;
        cin>>name;
        Class8 obj8(name);
        Class8* ptr=&obj8;
        cout<<((Class2 *)ptr)->Class2::Class1::GetName()<<endl;
        cout<<((Class4 *)ptr)->Class4::Class1::GetName()<<endl;
        cout<<((Class5 *)ptr)->Class5::Class1::GetName()<<endl;
        cout<<ptr->::Class2::GetName()<<endl;
        cout<<ptr->::Class3::GetName()<<endl;
        cout<<ptr->::Class4::GetName()<<endl;
        cout<<ptr->::Class5::GetName()<<endl;
        cout<<ptr->::Class6::GetName()<<endl;
        cout<<ptr->::Class7::GetName()<<endl;
        cout<<ptr->::Class8::GetName();
        return(0);
}
```

Тестирование

Результат тестирования программы представлен в следующей таблице.

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
Ident	Ident_1 Ident_1 Ident_1 Ident_2 Ident_3 Ident_4 Ident_5 Ident_6 Ident_7 Ident_8	Ident_1 Ident_1 Ident_1 Ident_2 Ident_3 Ident_4 Ident_5 Ident_6 Ident_7 Ident_8

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ (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_laboratornyh_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).