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 5_3_1 »

С тудент группы	ИКБО-27-21	Шевелёв И.А.
Руководитель практики	Ассистент	Морозов В.А.
Работа представлена	«___»_____ 2022 г.	_____
		(подпись студента)
Оценка		_____
		(подпись руководителя)

Москва 2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Постановка задачи

Полиморфизм в иерархии классов

Описать четыре класса которые последовательно наследуют друг друга, с номерами классов 1, 2, 3, 4. В каждом классе реализовать виртуальный метод с открытым доступом и одинаковым именем. Метод вычисляет значение многочлена степени номера класса и возвращает полученный результат. Коэффициенты и переменная многочлена целочисленные.

В основной функции реализовать алгоритм, в котором использовать один указатель на объект класса. Алгоритм:

1. Объявление указателя на объект класса.
2. Объявление четырех целочисленных переменных a1, a2, a3 a4, которые соответствуют коэффициентам многочлена (a1*x + a2*x*x + a3*x*x*x + a4*x*x*x*x).
3. Объявление целочисленной переменной x, которая соответствует переменной многочлена.
4. Ввод значения переменных a1, a2, a3 a4.
5. Создание объекта класса 4 посредством параметризированного конструктора, передав в качестве аргументов a1, a2, a3 a4. Обеспечить передачу необходимых коэффициентов объектам согласно наследственности классов.
6. Начало цикла
    1. Реализовать ввод значения переменной x.
    2. Если значение x равно нулю, то завершить цикл.
    3. Иначе, реализовать ввод значения номера класса.
    4. Согласно номеру класса вызвать метод вычисления многочлена посредством объекта, который соответствует номеру класса и

результат вывести.

7. Конец цикла.

Описание входных данных

Первая строка:

«целое число, значение a1»«целое число, значение a2»«целое число, значение a3»«целое число, значение a4»

Начиная со второй строки, построчно:

«целое число, значение x»«целое число, номер класса»

Описание выходных данных

Первая строка:

a1 = «целое число» a2 = «целое число» a3 = «целое число» a4 = «целое число»

Наименование коэффициента отделяется от предыдущего целого числа четырьмя пробелами.

Со второй строки и далее построчно:

Class «номер класса» F(«значение переменной x») = «значение многочлена»

Фрагменту « F(» предшествует 4 пробела

Метод решения

Основная программа:

Объекты ввода/вывода потока данных (cin/cout библиотеки <iostream>)

Целочисленные переменные

Объект класса class_4

Указатель

Оператор new, break continue

Цикл while

Условный оператор if

Указатель object на класс class_4

Класс class_1

Модификатор доступа public

Параметризированный конструктор

Метод get_count

Ключевое слово virtual

Объекты ввода/вывода потока данных (cin/cout библиотеки <iostream>)

Функция pow библиотеки <cmath>

Модификатор доступа protected:

Целочисленные переменные

Класс class_2

Модификатор доступа public

Параметризированный конструктор

Метод get_count

Ключевое слово virtual

Объекты ввода/вывода потока данных (cin/cout библиотеки <iostream>)

Функция pow библиотеки <cmath>

Класс class_3

Модификатор доступа public

Параметризированный конструктор

Метод get_count

Ключевое слово virtual

Объекты ввода/вывода потока данных (cin/cout библиотеки <iostream>)

Функция pow библиотеки <cmath>

Класс class_4

Модификатор доступа public

Параметризированный конструктор

Метод get_count

Ключевое слово virtual

Объекты ввода/вывода потока данных (cin/cout библиотеки <iostream>)

Функция pow библиотеки <cmath>

Описание алгоритма

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

Конструктор класса: class_1

Модификатор доступа: public

Функционал: Присвоение значений a1, a2, a3, a4

Параметры: int a1, int a2, int a3, int a4

Алгоритм конструктора представлен в таблице 1.

Таблица 1. Алгоритм конструктора класса class_1

№	Предикат	Действия	№ перехода	Комментарий
1		Присвоение свойству a1 текущего объекта передоваемое значение a1	2	
2		Присвоение свойству a2 текущего объекта передоваемое значение a2	3	
3		Присвоение свойству a3 текущего объекта передоваемое значение a3	4	
4		Присвоение свойству a4 текущего объекта передоваемое значение a4	∅	

Класс объекта: class_1

Модификатор доступа: public

Метод: get_count

Функционал: Считает многочлен с учётом переданного x

Параметры: int x

Возвращаемое значение: int, значение многолчена

Алгоритм метода представлен в таблице 2.

Таблица 2. Алгоритм метода get_count класса class_1

№	Предикат	Действия	№ перехода	Комментарий
1		Вывод "\nClass 1"	2	
2		Возращает значение a1*pow(x, 1)	∅	

Конструктор класса: class_2

Модификатор доступа: public

Функционал: Передаёт значение в наследуемый конструктор class_1

Параметры: int a1, int a2, int a3, int a4

Алгоритм конструктора представлен в таблице 3.

Таблица 3. Алгоритм конструктора класса class_2

№	Предикат	Действия	№ перехода	Комментарий
1			∅	

Класс объекта: class_2

Модификатор доступа: public

Метод: get_count

Функционал: Считает многочлен с учётом переданного x

Параметры: int x

Возвращаемое значение: int, значение многолчена

Алгоритм метода представлен в таблице 4.

Таблица 4. Алгоритм метода get_count класса class_2

№	Предикат	Действия	№ перехода	Комментарий
1		Вывод "\nClass 2"	2	
2		Возращает значение a1*pow(x, 1) + a2 * pow(x, 2)	∅	

Конструктор класса: class_3

Модификатор доступа: public

Функционал: Передаёт значение в наследуемый конструктор class_2

Параметры: int a1, int a2, int a3, int a4

Алгоритм конструктора представлен в таблице 5.

Таблица 5. Алгоритм конструктора класса class_3

№	Предикат	Действия	№ перехода	Комментарий
1			∅	

Класс объекта: class_3

Модификатор доступа: public

Метод: get_count

Функционал: Считает многочлен с учётом переданного x

Параметры: int x

Возвращаемое значение: int, значение многолчена

Алгоритм метода представлен в таблице 6.

Таблица 6. Алгоритм метода get_count класса class_3

№	Предикат	Действия	№ перехода	Комментарий
1		Вывод "\nClass 3"	2	
2		Возращает значение a1*pow(x, 1) + a2 * pow(x, 2) + a3 * pow(x, 3)	∅	

Конструктор класса: class_4

Модификатор доступа: public

Функционал: Передаёт значение в наследуемый конструктор class_3

Параметры: int a1, int a2, int a3, int a4

Алгоритм конструктора представлен в таблице 7.

Таблица 7. Алгоритм конструктора класса class_4

№	Предикат	Действия	№ перехода	Комментарий
1			∅	

Класс объекта: class_4

Модификатор доступа: public

Метод: get_count

Функционал: Считает многочлен с учётом переданного x

Параметры: int x

Возвращаемое значение: int, значение многолчена

Алгоритм метода представлен в таблице 8.

Таблица 8. Алгоритм метода get_count класса class_4

№	Предикат	Действия	№ перехода	Комментарий
1		Вывод "\nClass 4"	2	
2		Возращает значение a1*pow(x, 1) + a2 * pow(x, 2) + a3 * pow(x, 3) + a4 * pow(x, 4)	∅	

Функция: main

Функционал: Главная функция программы

Параметры: нет

Возвращаемое значение: int, Значение успеха выполнение программы

Алгоритм функции представлен в таблице 9.

Таблица 9. Алгоритм функции main

№	Предикат	Действия	№ перехода	Комментарий
1		Объявление указателя object на класс class_4	2	
2		Объявление целочисленных переменных a1, a2, a3, a4	3	
3		Объявление целочисленных переменных x, number, result	4	
4		Ввод значений a1, a2, a3, a4	5	
5		Присвоение указателю object созданному объекту класса class_4 с параметрами a1, a2, a3, a4	6	

6		Вывод "a1 = " a1 "a2 = " a2 "a3 = " a3 "a4 = " a4	7	
7	true		8	
			∅	
8		Ввод значение x	9	
9	x == 0	break	∅	
			10	
10		Ввод значение number	11	
11	number == 1	Присвоение переменной result значение метода get_count(x) класса class_1	12	
	number == 2	Присвоение переменной result значение метода get_count(x) класса class_2	12	
	number == 3	Присвоение переменной result значение метода get_count(x) класса class_3	12	
	number == 4	Присвоение переменной result значение метода get_count(x) класса class_4	12	
		continue	7	
12		Вывод "F(" x ") = " result	7	

Блок-схема алгоритма

Представим описание алгоритмов в графическом виде на рисунках ниже.

Рис. 1. Блок-схема алгоритма.

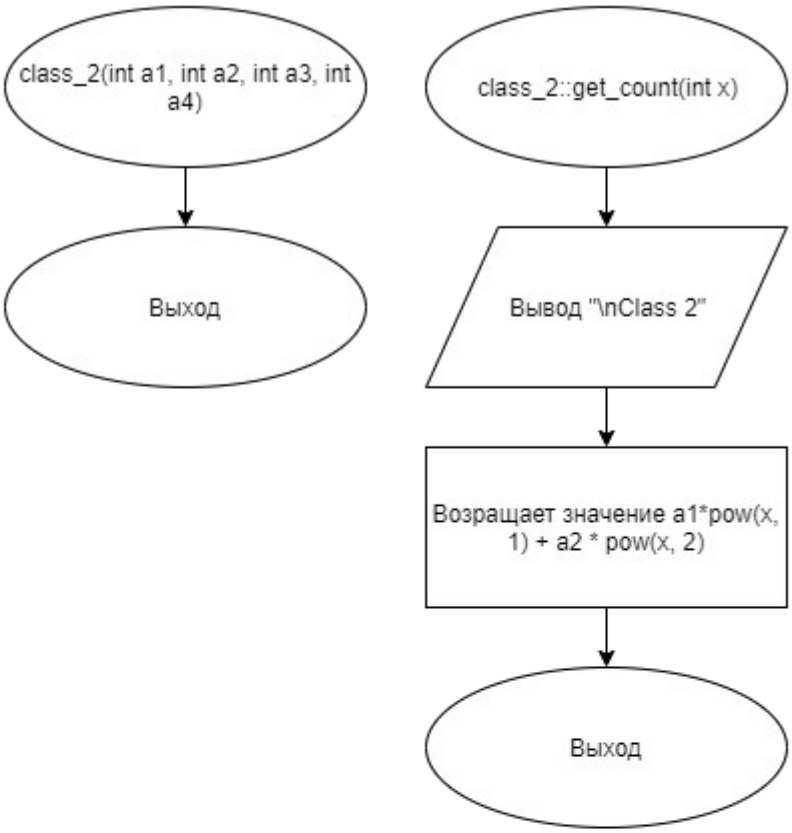

Рис. 2. Блок-схема алгоритма.

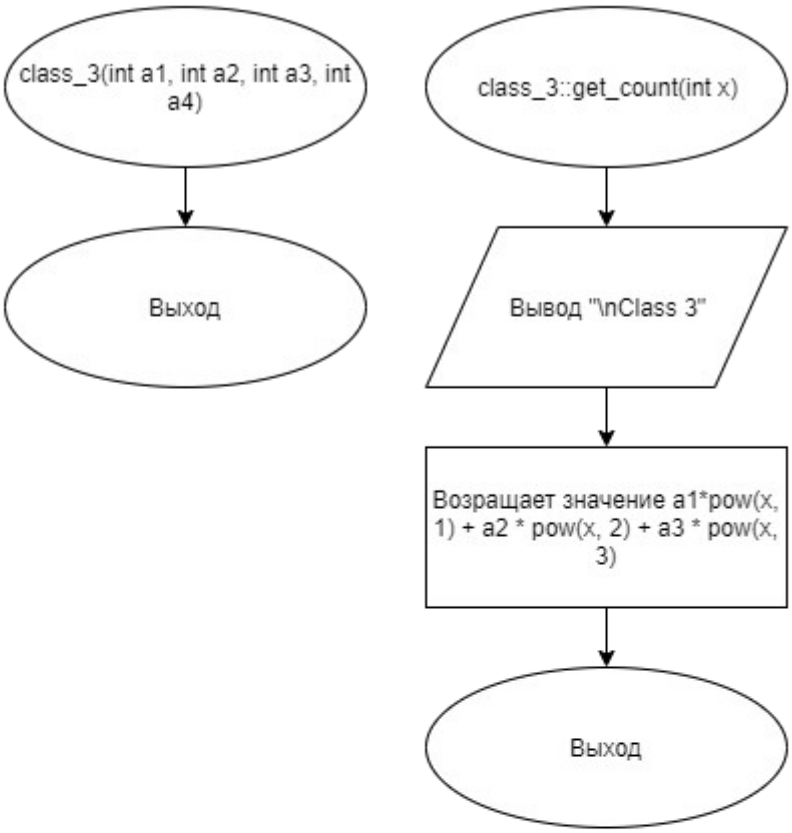

Рис. 3. Блок-схема алгоритма.

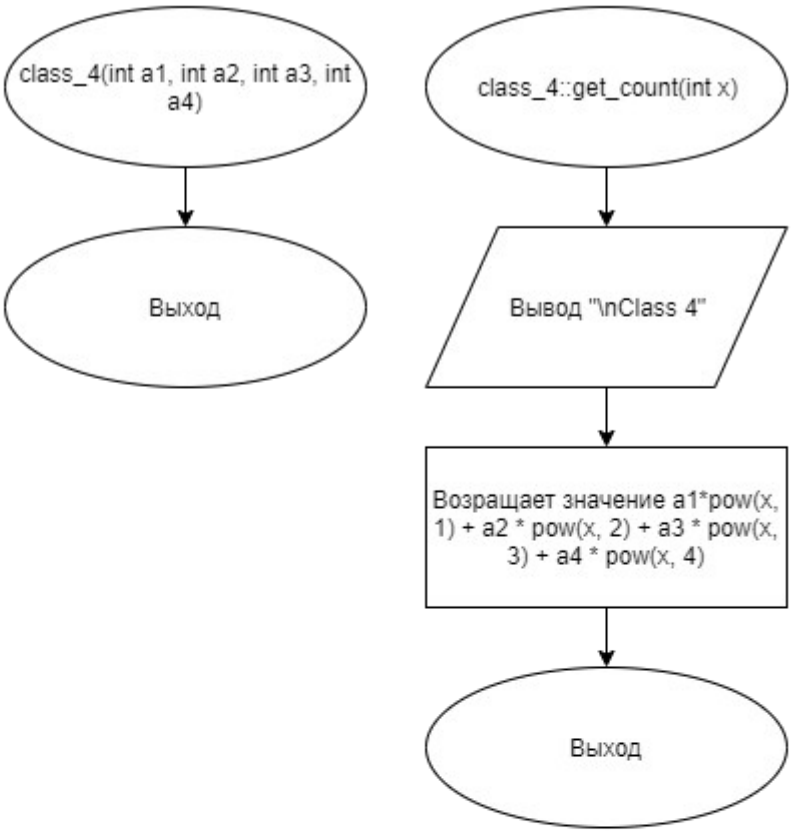

Рис. 4. Блок-схема алгоритма.

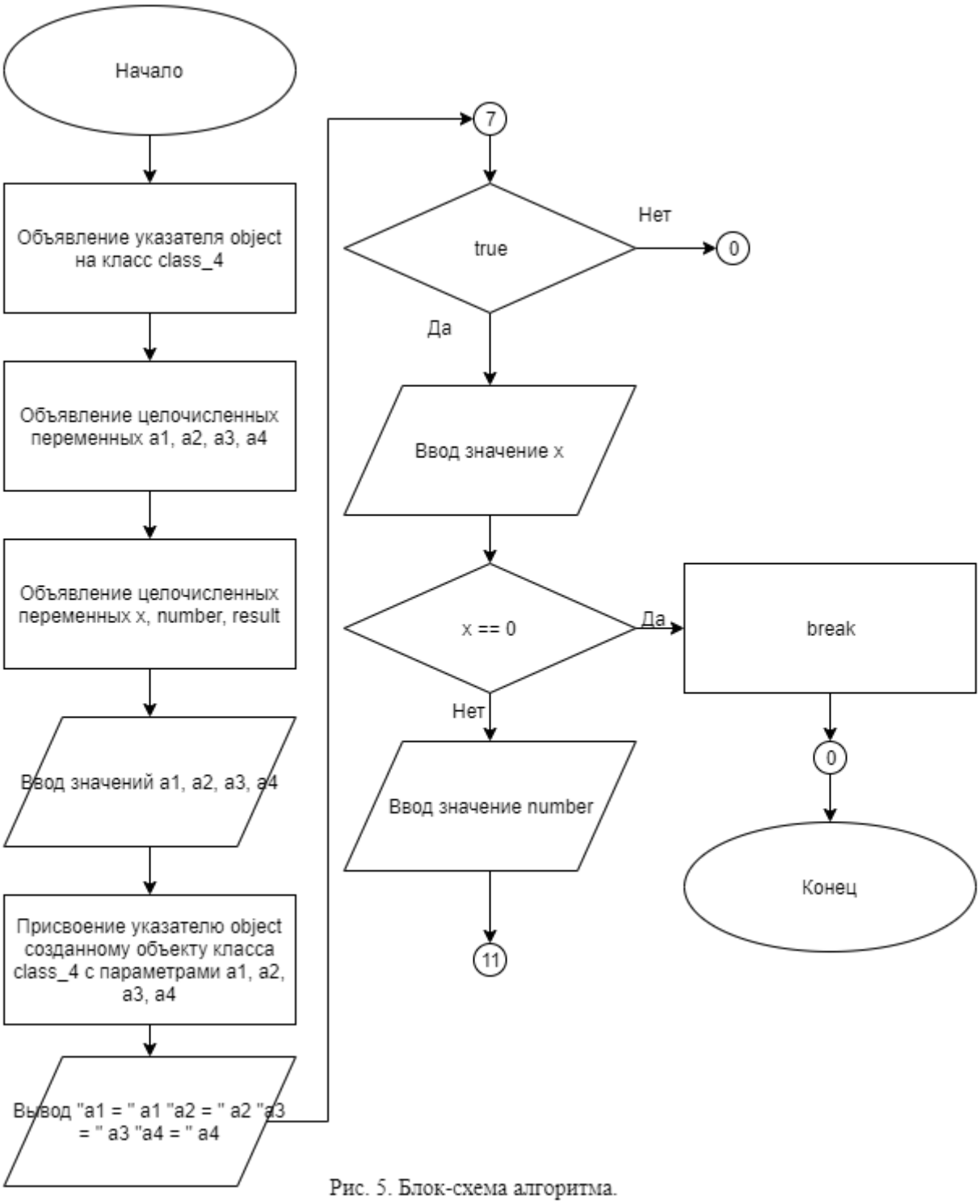

Рис. 5. Блок-схема алгоритма.

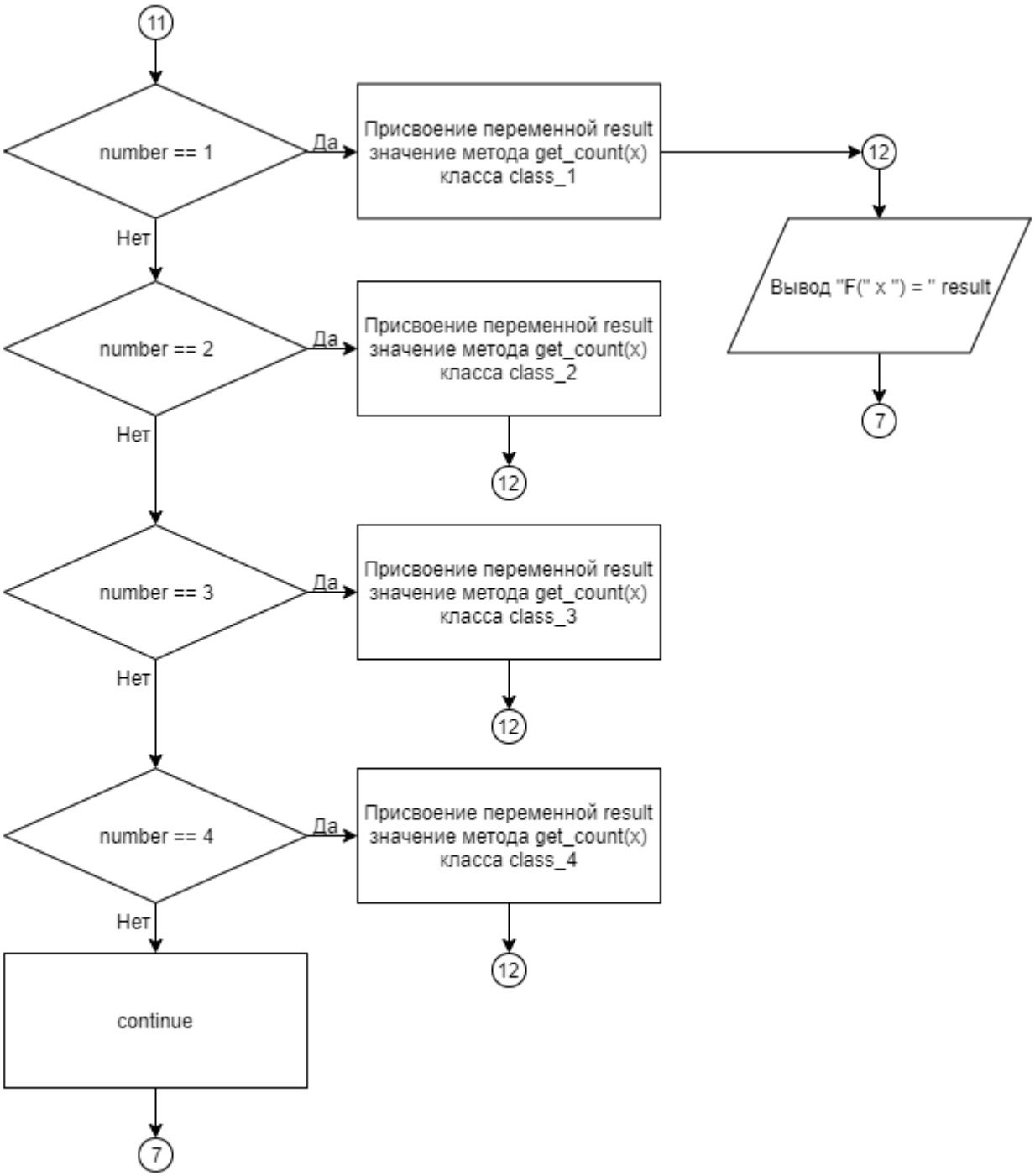

Рис. 6. Блок-схема алгоритма.

Код программы

Программная реализация алгоритмов для решения задачи представлена ниже.

Файл class_1.cpp

```
#include "class_1.h"

class_1::class_1(int a1, int a2, int a3, int a4){
        this->a1 = a1;
        this->a2 = a2;
        this->a3 = a3;
        this->a4 = a4;
}
int class_1::get_count(int x){
        cout << "\nClass 1";
        return a1 * pow(x, 1);
}
```

Файл class_1.h

```
#ifndef CLASS1_H
#define CLASS1_H
#include <iostream>
#include <cmath>
#include <string>
using namespace std;

class class_1{
public:
        class_1(int a1, int a2, int a3, int a4);
        virtual int get_count(int x);
protected:
        int a1, a2, a3, a4;
};
#endif
```

Файл class_2.cpp

```
#include "class_2.h"
```

```
class_2::class_2(int a1, int a2, int a3, int a4) : class_1(a1, a2, a3, a4){}
int class_2::get_count(int x){
        cout << "\nClass 2";
        return a1 * pow(x, 1) + a2 * pow(x, 2);
}
```

Файл class_2.h

```
#ifndef CLASS2_H
#define CLASS2_H
#include "class_1.h"
class class_2 : public class_1{
public:
        class_2(int a1, int a2, int a3, int a4);
        virtual int get_count(int x);
};
#endif
```

Файл class_3.cpp

```
#include "class_3.h"

class_3::class_3(int a1, int a2, int a3, int a4) : class_2(a1, a2, a3, a4){}
int class_3::get_count(int x){
        cout << "\nClass 3";
        return a1 * pow(x, 1) + a2 * pow(x, 2) + a3 * pow(x, 3);
}
```

Файл class_3.h

```
#ifndef CLASS3_H
#define CLASS3_H
#include "class_2.h"
class class_3 : public class_2{
public:
        class_3(int a1, int a2, int a3, int a4);
        virtual int get_count(int x);
};
#endif
```

Файл class_4.cpp

```
#include "class_4.h"
```

```
class_4::class_4(int a1, int a2, int a3, int a4) : class_3(a1, a2, a3, a4){}
int class_4::get_count(int x){
        cout << "\nClass 4";
        return a1 * pow(x, 1) + a2 * pow(x, 2) + a3 * pow(x, 3) + a4 * pow(x,
4);
}
```

Файл class_4.h

```
#ifndef CLASS4_H
#define CLASS4_H
#include "class_3.h"
class class_4 : public class_3{
public:
        class_4(int a1, int a2, int a3, int a4);
        virtual int get_count(int x);
};
#endif
```

Файл main.cpp

```
#include <stdlib.h>
#include <stdio.h>
#include "class_4.h"

int main()
{
        // program here
        class_4* object;
        int a1, a2, a3, a4;
        int x, number, result;
        cin >> a1 >> a2 >> a3 >> a4;
        object = new class_4(a1, a2, a3, a4);
        cout << "a1 = " << a1;
        cout << "    " << "a2 = " << a2;
        cout << "    " << "a3 = " << a3;
        cout << "    " << "a4 = " << a4;

        while(true){
                cin >> x;
                if(x == 0)
                        break;
                else{
                        cin >> number;
                        if(number == 1)
                                result = object->class_1::get_count(x);
```

```
                        else if(number == 2)
                               result = object->class_2::get_count(x);
                        else if(number == 3)
                               result = object->class_3::get_count(x);
                        else if(number == 4)
                               result = object->class_4::get_count(x);
                        else continue;
                        cout << "    F( " << x << " ) = " << result;
                }
        }
        return(0);
}
```

Тестирование

Результат тестирования программы представлен в следующей таблице.

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
12 14 16 18 2 1 2 2 2 3 2 4 2 5 2 6 2 2 0	a1 = 12 a2 = 14 a3 = 16 a4 = 18 Class 1 F(2) = 24 Class 2 F(2) = 80 Class 3 F(2) = 208 Class 4 F(2) = 496 Class 2 F(2) = 80	a1 = 12 a2 = 14 a3 = 16 a4 = 18 Class 1 F(2) = 24 Class 2 F(2) = 80 Class 3 F(2) = 208 Class 4 F(2) = 496 Class 2 F(2) = 80
1 2 3 4 10 1 10 2 10 3 10 4 10 5 0	a1 = 1 a2 = 2 a3 = 3 a4 = 4 Class 1 F(10) = 10 Class 2 F(10) = 210 Class 3 F(10) = 3210 Class 4 F(10) = 43210	a1 = 1 a2 = 2 a3 = 3 a4 = 4 Class 1 F(10) = 10 Class 2 F(10) = 210 Class 3 F(10) = 3210 Class 4 F(10) = 43210
7 2 3 4 9 1 0	a1 = 7 a2 = 2 a3 = 3 a4 = 4 Class 1 F(9) = 63	a1 = 7 a2 = 2 a3 = 3 a4 = 4 Class 1 F(9) = 63

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).