Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

Разработать систему, которая демонстрирует реализацию принципа полиморфизма при иерархическом конструировании объекта.

Спроектировать 4 разных объекта. Перенумеровать классы их принадлежности от 1 до 4. Каждый объект имеет свойство целочисленного типа в закрытом доступе. Это свойство хранит значение коэффициента многочлена, соответствующее номеру класса. Его значение определяется в конструкторе объекта, посредством значений параметра целочисленного типа.

У каждого объекта есть метод в открытом доступе, с одинаковым наименованием. У этого метода есть один целочисленный параметр, который содержит значение переменной многочлена. Метод вычисляет значение многочлена степени согласно номеру класса принадлежности объекта и возвращает полученный целочисленный результат.

Сконструировать иерархию вложенных объектов (наследственность объектов). Объект второго класса содержит в своем составе объект первого класса. Объект третьего класса содержит в своем составе объект второго класса. Объект четвертого класса содержит в своем составе объект третьего класса. Обеспечить передачу необходимых коэффициентов конструкторам объектов согласно наследственности.

Алгоритм конструирования и отработки системы:

1. Объявляется указатель на объект **четвертого класса.**

2. Объявляются четыре целочисленные переменные a1, a2, a3 a4, значения которых соответствуют коэффициентам многочлена (a1*x + a2*x*x + a3*x*x*x + a4*x*x*x*x).

3. Объявляется целочисленная переменная x, для хранения значения

переменной многочлена.

4. Объявляется целочисленная переменная i_class, для хранения значения номера класса.

5. Вводятся значения переменных a1, a2, a3, a4.

6. Создается объект класс 4 посредством параметризированного конструктора, которому передаются в качестве аргументов a1, a2, a3 a4. Адрес объекта присваивается объявленному указателю.

7. Начало цикла

7.1. Вводится значение переменной x.

7.2. Если значение x равно нулю, то цикл завершается.

7.3. Иначе, вводится значение номера класса.

7.4. Согласно номеру класса, вызывается метод вычисления многочлена посредством объекта, который соответствует номеру класса и полученный результат выводится.

8. Конец цикла.

9. Завершается работа системы.

1.1 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»`

1.2 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

2 МЕТОД РЕШЕНИЯ

Для решения задачи используется:

- объект cl4 класса Class4 предназначен для ;
- библиотека iostream;
- объекты стандартного потока ввода/вывода данных cin/cout;
- условная конструкция if...else;
- цикл с предусловием while;
- арифметические операторы.

Класс Class1:

- свойства/поля:
    - поле коэффициент многочлена:
        - наименование — k1;
        - тип — int;
        - модификатор доступа — private;
- функционал:
    - метод calculate — считает значение многочлена степени 1;
    - метод Class1 — конструктор.

Класс Class2:

- свойства/поля:
    - поле коэффициент многочлена:
        - наименование — k2;
        - тип — int;
        - модификатор доступа — private;
- функционал:
    - метод calculate — считает значение многочлена степени 2;
    - метод Class2 — конструктор.

Класс Class3:

- свойства/поля:
  - поле коэффициент многочлена:
    - наименование — k3;
    - тип — int;
    - модификатор доступа — private;
- функционал:
  - метод calculate — считает значение многочлена степени 3;
  - метод Class3 — конструктор.

Класс Class4:

- свойства/поля:
  - поле коэффициент многочлена:
    - наименование — k4;
    - тип — int;
    - модификатор доступа — private;
- функционал:
  - метод calculate — считает значение многочлена степени 4;
  - метод Class4 — конструктор.

Таблица 1 – Иерархия наследования классов

№	Имя класса	Классы-наследники	Модификатор доступа при наследовании	Описание	Номер
1	Class1				
		Class2	public		2
2	Class2				
		Class3	public		3
3	Class3				
		Class4	public		4

№	Имя класса	Классы-наследники	Модификатор доступа при наследовании	Описание	Номер
4	Class4				

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 метода calculate класса Class1

Функционал: считает значение многочлена степени 1.

Параметры: целочисленный параметр x.

Возвращаемое значение: целочисленное значение - результат вычислений.

Алгоритм метода представлен в таблице 2.

Таблица 2 – Алгоритм метода calculate класса Class1

№	Предикат	Действия	№ перехода
1		Возвращение k1*x	∅

3.2 Алгоритм конструктора класса Class1

Функционал: конструктор.

Параметры: целочисленный параметр p1.

Алгоритм конструктора представлен в таблице 3.

Таблица 3 – Алгоритм конструктора класса Class1

№	Предикат	Действия	№ перехода
1		Присвоение полю k1 значения p1	∅

3.3 Алгоритм метода calculate класса Class2

Функционал: считает значение многочлена степени 2.

Параметры: целочисленный параметр x.

Возвращаемое значение: целочисленное значение - результат вычислений.

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода calculate класса Class2

№	Предикат	Действия	№ перехода
1		Возврат суммы k2*x*x и возвращаемого значения метода calculate родительского объекта	∅

3.4 Алгоритм конструктора класса Class2

Функционал: конструктор.

Параметры: целочисленные параметры p1, p2.

Алгоритм конструктора представлен в таблице 5.

Таблица 5 – Алгоритм конструктора класса Class2

№	Предикат	Действия	№ перехода
1		Вызов конструктора родителя с передачей в него параметров p1, p2	2
2		Присвоение полю k2 значения p2	∅

3.5 Алгоритм метода calculate класса Class3

Функционал: считает значение многочлена степени 3.

Параметры: целочисленный параметр x.

Возвращаемое значение: целочисленное значение - результат вычислений.

Алгоритм метода представлен в таблице 6.

Таблица 6 – Алгоритм метода calculate класса Class3

№	Предикат	Действия	№ перехода
1		Возврат суммы k3*x*x*x и возвращаемого значения метода calculate	∅

№	Предикат	Действия	№ перехода
		родительского объекта	

3.6 Алгоритм конструктора класса Class3

Функционал: конструктор.

Параметры: целочисленные параметры p1, p2, p3.

Алгоритм конструктора представлен в таблице 7.

Таблица 7 – Алгоритм конструктора класса Class3

№	Предикат	Действия	№ перехода
1		Вызов конструктора родителя с передачей в него параметров p1, p2, p3	2
2		Присвоение полю k3 значения p3	∅

3.7 Алгоритм метода calculate класса Class4

Функционал: считает значение многочлена степени 4.

Параметры: целочисленный параметр x.

Возвращаемое значение: целочисленное значение - результат вычислений.

Алгоритм метода представлен в таблице 8.

Таблица 8 – Алгоритм метода calculate класса Class4

№	Предикат	Действия	№ перехода
1		Возврат суммы k4*x*x*x*x и возвращаемого значения метода calculate родительского объекта	∅

3.8 Алгоритм конструктора класса Class4

Функционал: конструктор.

Параметры: целочисленные параметры p1, p2, p3, p4.

Алгоритм конструктора представлен в таблице 9.

Таблица 9 – Алгоритм конструктора класса Class4

№	Предикат	Действия	№ перехода
1		Вызов конструктора родителя с передачей в него параметров p1, p2, p3, p4	2
2		Присвоение полю k4 значения p4	∅

3.9 Алгоритм функции main

Функционал: главная функция программы.

Параметры: нет.

Возвращаемое значение: целое, идентификатор работоспособности программы.

Алгоритм функции представлен в таблице 10.

Таблица 10 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		Объявление указателя cl4 на объект класса Class4	2
2		Объявление целочисленных переменных a1, a2, a3, a4, i_data, x	3
3		Ввод значений a1, a2, a3, a4 с помощью cin	4
4		Создание объекта класса Class4 с передачей в конструктор параметров a1, a2, a3, a4 и присвоение указателю на этот объект переменной cl4	5
5		Вывод строки "a1 = ", a1, "a2 = ", a2, "a3 = ", a3, "a4 = ", a4 и переход на новую строку	6
6		Ввод значения x	7

№	Предикат	Действия	№ перехода
7	x == 0?		∅
		Ввод значения i_class	8
8		Вывод строки "Class", i_class, " ", "F(", x, ")"	9
9	i_class == 1?	Вызов метода calculate с передачей в него параметра x у объекта класса Class1, который является родительским объектом класса Class2, который является родительским объектом объекта класса Class3, который является родительским объектом cl4	13
			10
10	i_class == 2?	Вызов метода calculate с передачей в него параметра x у объекта класса Class2, который является родительским объектом объекта класса Class3, который является родительским объектом cl4	13
			11
11	i_class == 3?	Вызов метода calculate с передачей в него параметра x у объекта класса Class3, который является родительским объектом cl4	13
			12
12	i_class == 4?	Вызов метода calculate с передачей в него параметра x у объекта cl4	13
			∅
13		Переход на новую строку	6

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-7.

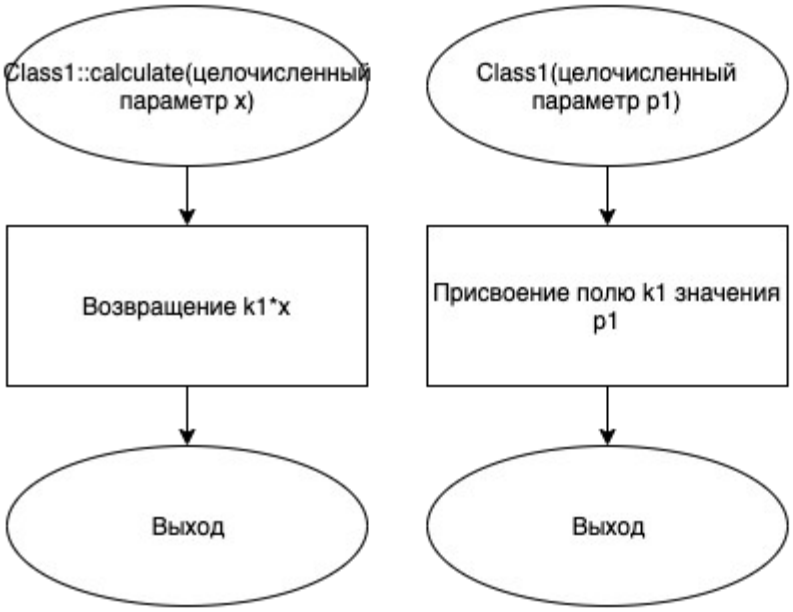

Рисунок 1 – Блок-схема алгоритма

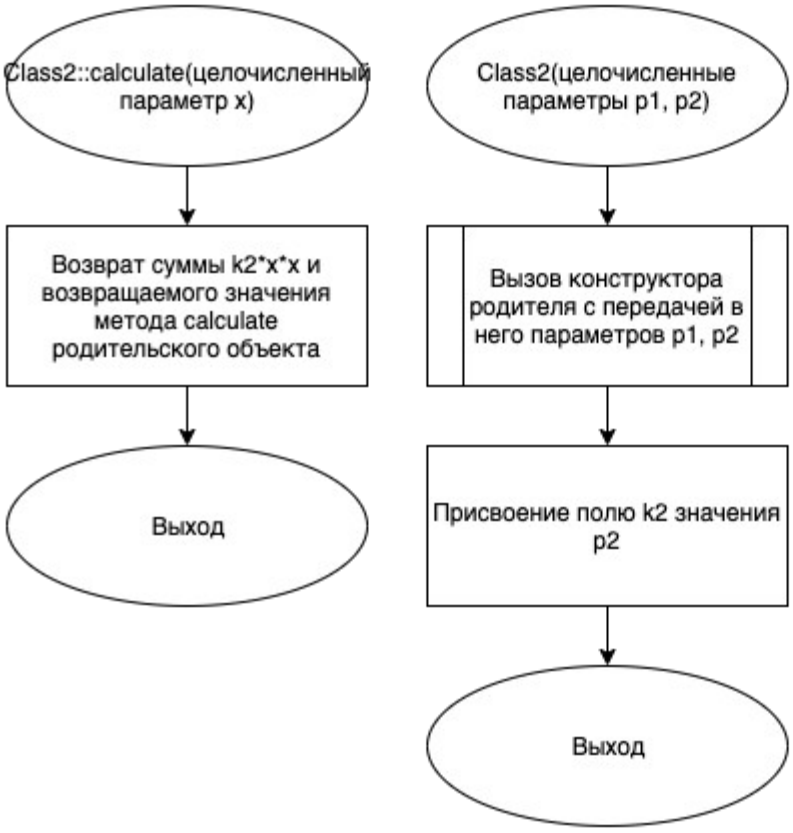

Рисунок 2 – Блок-схема алгоритма

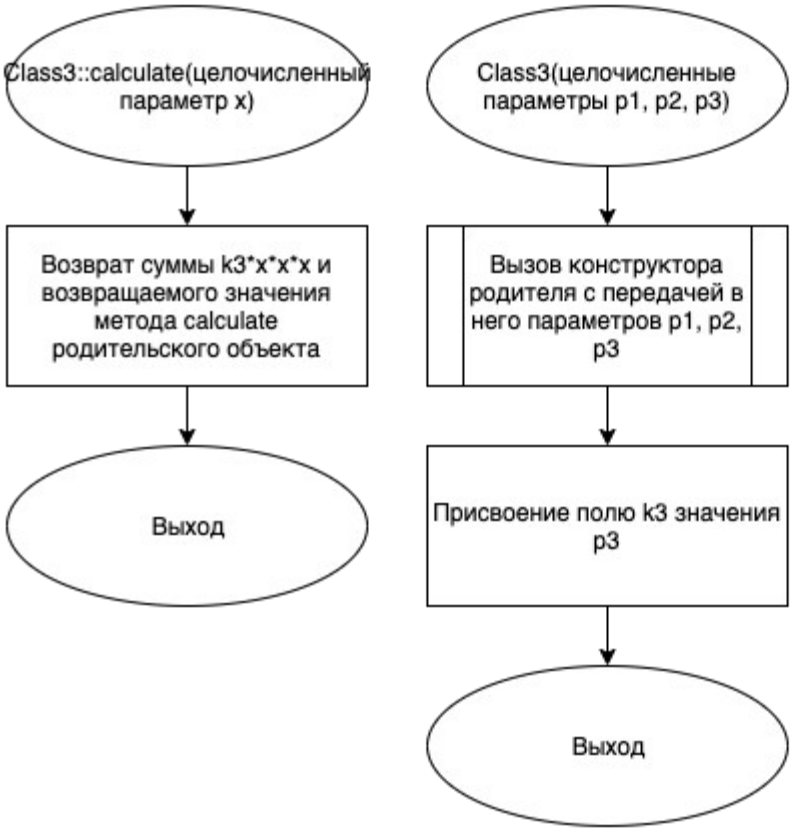

Рисунок 3 – Блок-схема алгоритма

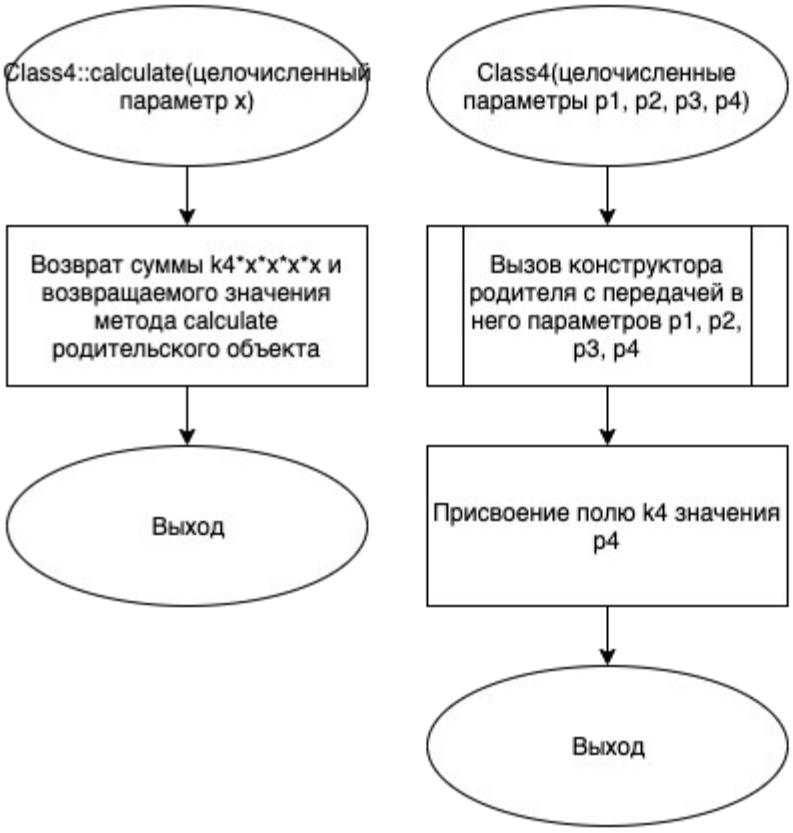

Рисунок 4 – Блок-схема алгоритма

Рисунок 5 – Блок-схема алгоритма

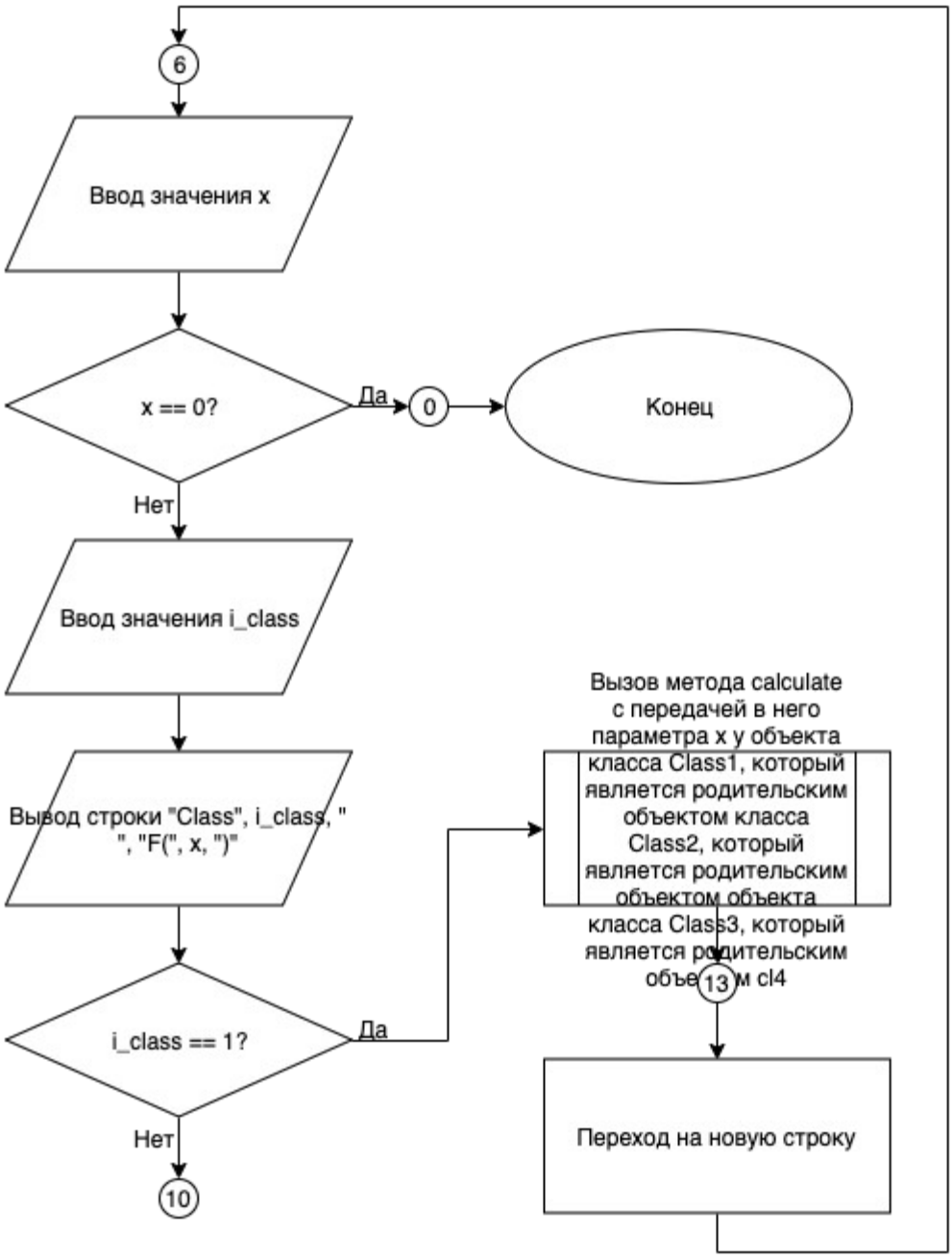

Рисунок 6 – Блок-схема алгоритма

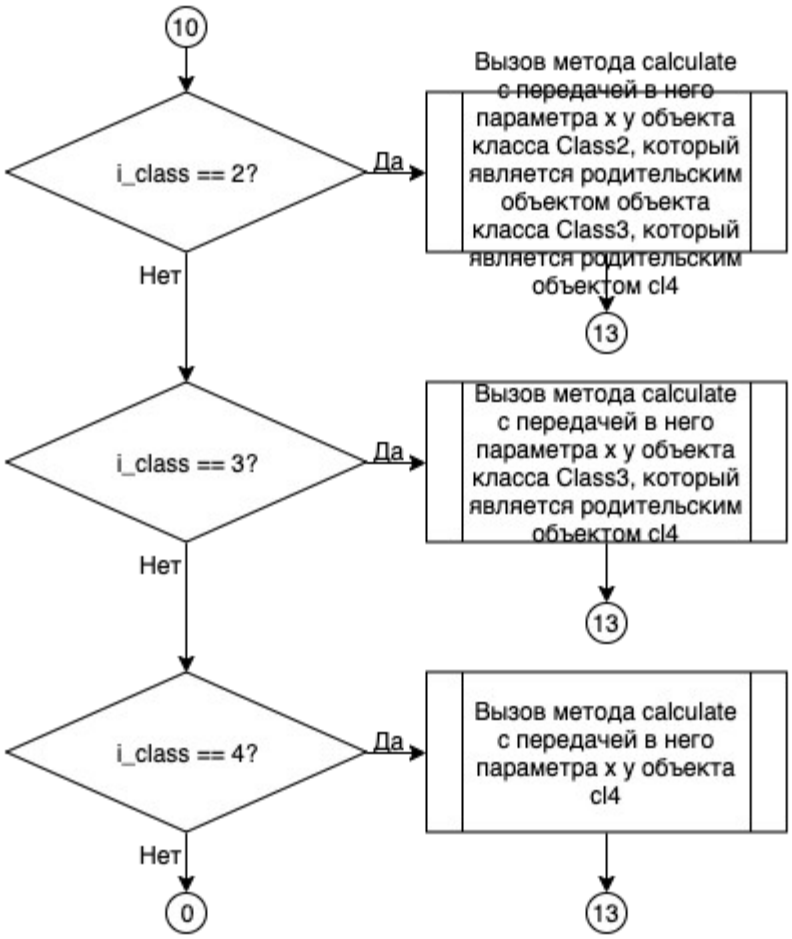

Рисунок 7 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл Class1.cpp

Листинг 1 – Class1.cpp

```
#include "Class1.h"
#include <iostream>
using namespace std;

int Class1::calculate(int x){
   return k1*x;
}

Class1::Class1(int p1){
   k1 = p1;
}
```

5.2 Файл Class1.h

Листинг 2 – Class1.h

```
#ifndef __CLASS1__H
#define __CLASS1__H

class Class1 {
   int k1;
public:
   int calculate(int p1);
   Class1(int a1);
};

#endif
```

5.3 Файл Class2.cpp

Листинг 3 – Class2.cpp

```
#include "Class2.h"
#include "Class1.h"
#include <iostream>
using namespace std;

int Class2::calculate(int x){
   return Class1::calculate(x) + k2*x*x;
}

Class2::Class2(int a1, int a2) : Class1(a1){
   k2 = a2;
}
```

5.4 Файл Class2.h

Листинг 4 – Class2.h

```
#ifndef __CLASS2__H
#define __CLASS2__H
#include "Class1.h"

class Class2: public Class1 {
   int k2;
public:
   int calculate(int x);
   Class2(int a1, int a2);
};

#endif
```

5.5 Файл Class3.cpp

Листинг 5 – Class3.cpp

```
#include "Class3.h"
#include "Class2.h"
#include "Class1.h"
#include <iostream>
using namespace std;
```

```
int Class3::calculate(int x){
   return Class2::calculate(x) + k3 *x *x *x;
}

Class3::Class3(int a1, int a2, int a3) : Class2(a1, a2){
   k3 = a3;
}
```

5.6 Файл Class3.h

Листинг 6 – Class3.h

```
#ifndef __CLASS3__H
#define __CLASS3__H
#include "Class2.h"

class Class3: public Class2 {
   int k3;
public:
   int calculate(int x);
   Class3(int a1, int a2, int a3);
};

#endif
```

5.7 Файл Class4.cpp

Листинг 7 – Class4.cpp

```
#include "Class4.h"
#include "Class3.h"
#include "Class2.h"
#include "Class1.h"
#include <iostream>
using namespace std;

int Class4::calculate(int x){
   return Class3::calculate(x) + k4 *x *x *x *x;
}

Class4::Class4(int a1, int a2, int a3, int a4) : Class3(a1, a2, a3){
   k4 = a4;
}
```

5.8 Файл Class4.h

Листинг 8 – Class4.h

```
#ifndef __CLASS4__H
#define __CLASS4__H
#include "Class3.h"

class Class4: public Class3 {
   int k4;
public:
   int calculate(int x);
   Class4(int a1, int a2, int a3, int a4);
};

#endif
```

5.9 Файл main.cpp

Листинг 9 – main.cpp

```
#include <iostream>
#include "Class1.h"
#include "Class2.h"
#include "Class3.h"
#include "Class4.h"
using namespace std;

int main()
{
   Class4* cl4;
   int a1, a2, a3, a4, x, i_class;

   cin >> a1 >> a2 >> a3 >> a4;
   cl4 = new Class4(a1, a2, a3, a4);

   cout << "a1 = " << a1 << "    a2 = " << a2 << "    a3 = " << a3 << "    a4
= " << a4 << endl;

   x = -1;
   while (true) {
      cin >> x;
      if (x == 0){
         break;
      } else {
         cin >> i_class;
         cout << "Class " << i_class << "    " << "F( " << x << " ) = ";
         switch (i_class){
```

```
                case 1:
                   cout << cl4->Class3::Class2::Class1::calculate(x);
                   break;
                case 2:
                   cout << cl4->Class3::Class2::calculate(x);
                   break;
                case 3:
                   cout << cl4->Class3::calculate(x);
                   break;
                case 4:
                   cout << cl4->calculate(x);
                   break;
             }
             cout << endl;
          }
       }
       return(0);
    }
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 11.

Таблица 11 – Результат тестирования программы

Входные данные	**Ожидаемые выходные данные**	**Фактические выходные данные**
1 1 1 1 0	a1 = 1 a2 = 1 a3 = 1 a4 = 1	a1 = 1 a2 = 1 a3 = 1 a4 = 1

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 19 Единая система программной документации.

2. Методическое пособие студента для выполнения 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).

3. 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).

4. Шилдт Г. C++: базовый курс. 3-е изд. Пер. с англ.. — М.: Вильямс, 2019. — 624 с.

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).