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 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

   6.1. Реализовать ввод значения переменной x.

   6.2. Если значение x равно нулю, то завершить цикл.

   6.3. Иначе, реализовать ввод значения номера класса.

   6.4. 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

№	название класса	Классы-наследники	Модификатор доступа	описание
1	Class1			Базовый класс
		Class2	public	
2	Class2			Класс, наследующийся от Class1
		Class3	public	
3	Class3			Класс, наследующийся от Class2
		Class4	public	
4	Class4			Класс, наследующийся от Class3

Список используемых объектов

- Объект object класса Class4

Структрура классов

- Класс Class1
  - Поля
    - поле для коэффицента a1
      - имя - a1
      - тип - int
      - модификатор доступа - protected
    - поле для коэффицента a2
      - имя - a1
      - тип - int
      - модификатор доступа - protected
    - поле для коэффицента a3
      - имя - a1
      - тип - int
      - модификатор доступа - protected
    - поле для коэффицента a4
      - имя - a1
      - тип - int
      - модификатор доступа - protected
  - Методы
    - Конструктор Class1
      - Функционал - Параметризированный конструктор с коэффицентами многочлена
    - метод get_count
      - Функционал - считает значение многочлена 1 степени - a1*x
- Класс Class2, наследуемый от Class1
  - Методы
    - Конструктор Class2
      - Функционал - наследуется от Class1
    - метод get_count
      - Функционал - считает значение многочлена 2 степени - a1*x + a2*x*x
- Класс Class3, наследуемый от Class2
  - Методы
    - Конструктор Class3
      - Функционал - наследуется от Class2
    - метод get_count
      - Функционал - считает значение многочлена 3 степени - a1*x + a2*x*x + a3*x*x*x
- Класс Class4, наследуемый от Class3
  - Методы
    - Конструктор Class4
      - Фунционал - наследуется от Class3
    - метод get_count

- Функционал-считает значение многочлена 4 степени- a1*x+ a2*x*x + a3*x*x*x + a4*x*x*x*x

Описание алгоритма

Функция: main

Функционал: главная функция программы

Параметры: -

Возвращаемое значение: 0, тип int

№	Предикат	Действия	№ перехода	Комментарий
1		вызов функции app без параметров	∅	в функции app реализован основной алгоритм

Функция: app

Функционал: В функции реализован основной алгоритм программы

Параметры: -

Возвращаемое значение: -

№	Предикат	Действия	№ перехода	Комментарий
1		Объявление указателя object на объект класса Class4	2	
2		Объявление переменных целого типа: a1,a2,a3,a4,x,number,result	3	
3		ввод a1,a2a,a3,a4	4	
4		Создание объекта класса Class4 с помощью контструктора,в который передаются коэффициенты многочлена a1,a2,a3,a4	5	
5		Вывод коэффицентов многочлена	6	
6		ввод x и номера класса	7	начало бесконечного

				цикла
7	x равен 0	-	∅	
	x не равен 0	-	8	
8	number равен 4	вызов метода get_count с параметром x	9	
	number равен 3	вызов метода Class3::get_count с параметром x	9	
	number равен 2	вызов метода Class2::get_count с параметром x	9	
	number равен 1	вызов метода Class1::get_count с параметром x	9	
	number другой	-	6	
9		присвоение в result результата метода	10	
10		Вывод f(x) = result	6	

Класс объекта: Class1

Модификатор доступа: public

Метод: Class1

Функционал: устанавливает коэффициенты многочлена

Параметры: a1,a2,a3,a4 - все целого типа - коэффициенты многочлена

Возвращаемое значение: -

№	Предикат	Действия	№ перехода	Комментарий
1		присвание в свойство a1 значения a1 из параметра	2	
2		присвание в свойство a2 значения a2 из параметра	3	
3		присвание в свойство a3 значения a3 из параметра	4	
4		присвание в свойство a4 значения a4 из параметра	∅	

Класс объекта: Class2

Модификатор доступа: public

Метод: Class2

Функционал: устанавливает коэффициенты многочлена

Параметры: a1,a2,a3,a4 - все целого типа - коэффициенты многочлена

Возвращаемое значение: -

№	Предикат	Действия	№ перехода	Комментарий
1		Вызов констуктора Class1 со всеми параметрами	∅	

Класс объекта: Class3

Модификатор доступа: public

Метод: Class3

Функционал: устанавливает коэффиценты многочлена

Параметры: a1,a2,a3,a4 - все целого типа - коэффиценты многочлена

Возвращаемое значение: -

№	Предикат	Действия	№ перехода	Комментарий
1		Вызов констуктора Class2 со всеми параметрами	∅	

Класс объекта: Class4

Модификатор доступа: public

Метод: Class4

Функционал: устанавливает коэффиценты многочлена

Параметры: a1,a2,a3,a4 - все целого типа - коэффиценты многочлена

Возвращаемое значение: -

№	Предикат	Действия	№ перехода	Комментарий
1		Вызов констуктора Class3 со всеми параметрами	∅	

Класс объекта: Class1

Модификатор доступа: public

Метод: Class1

Функционал: Считает многочлен с учетом переданного x

Параметры: int x - число, которое подставляет в многочлен

Возвращаемое значение: тип int - значение многочлена

№	Предикат	Действия	№ перехода	Комментарий
1		вывод "\nClass 1"	2	
2		возврат a1*x	∅	выражение степени 1, так как номер класса 1

Класс объекта: Class2

Модификатор доступа: public

Метод: Class2

Функционал: Считает многочлен с учетом переданного x

Параметры: int x - число, которое подставляет в многочлен

Возвращаемое значение: тип int - значение многочлена

№	Предикат	Действия	№ перехода	Комментарий
1		вывод "\nClass 2"	2	
2		возврат a1*x + a2*x*x	∅	выражение степени 2, так как номер класса 2

Класс объекта: Class3

Модификатор доступа: public

Метод: Class3

Функционал: Считает многочлен с учетом переданного x

Параметры: int x - число, которое подставляет в многочлен

Возвращаемое значение: тип int - значение многочлена

№	Предикат	Действия	№ перехода	Комментарий
1		вывод "\nClass 3"	2	
2		возврат a1*x + a2*x*x + a3*x*x*x	∅	выражение степени 3, так как номер класса 3

Класс объекта: Class4

Модификатор доступа: public

Метод: Class4

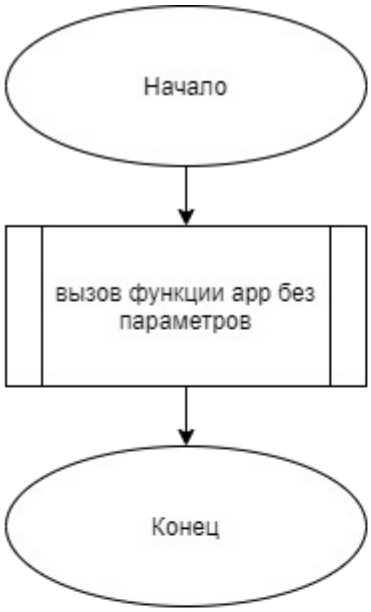
Функционал: Считает многочлен с учетом переданного x

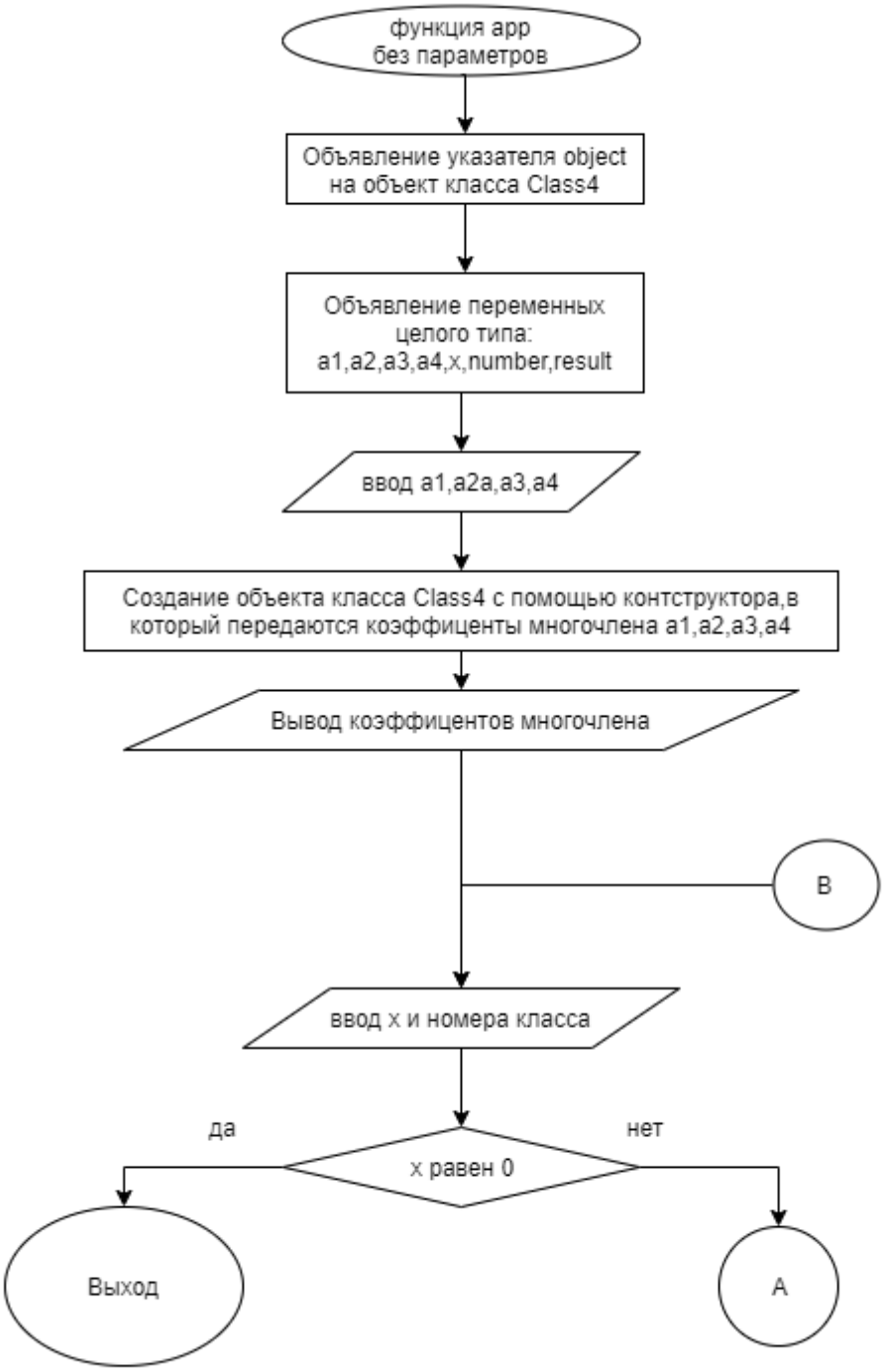
Параметры: int x - число, которое подставляет в многочлен

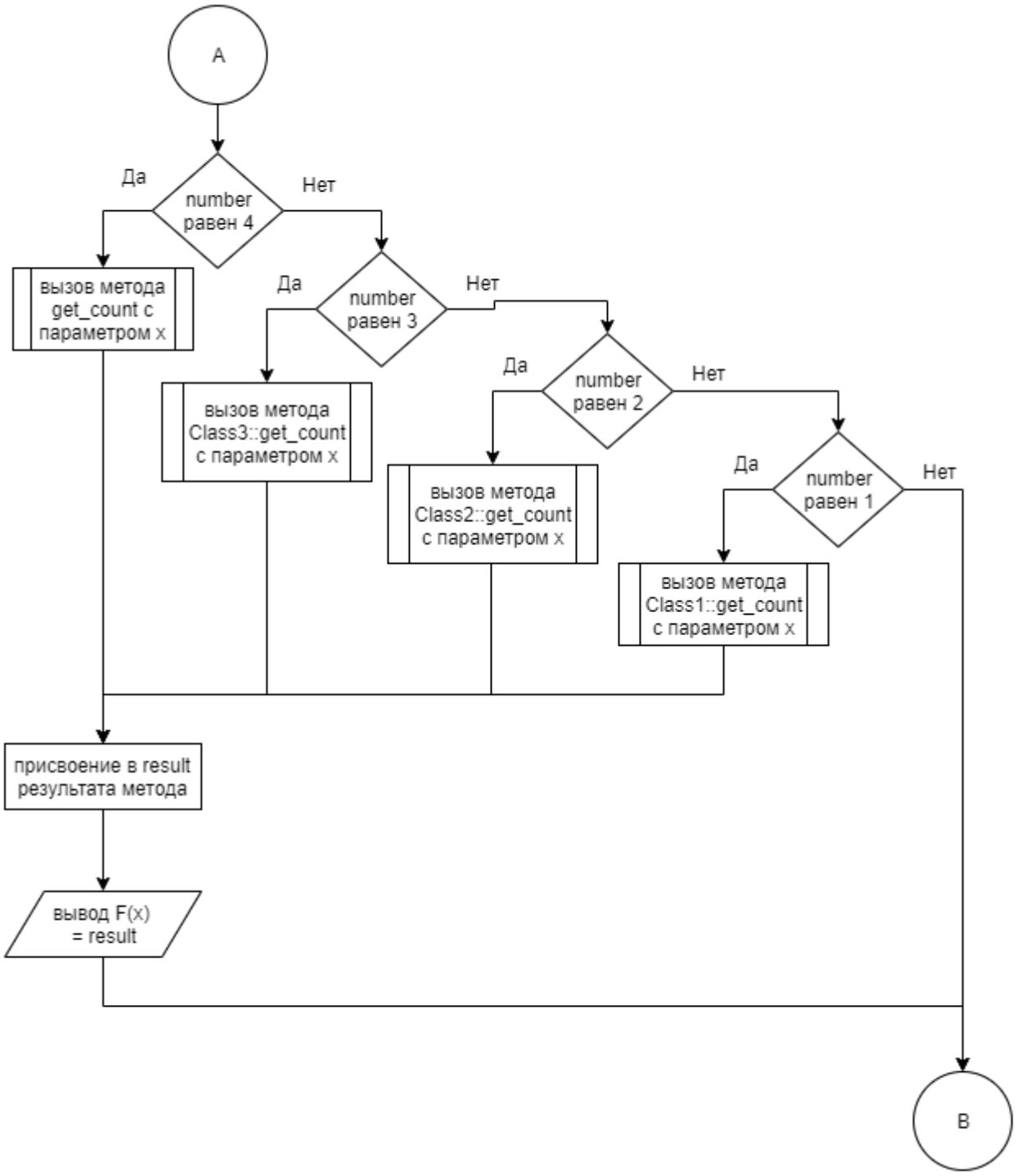
Возвращаемое значение: тип int - значение многочлена

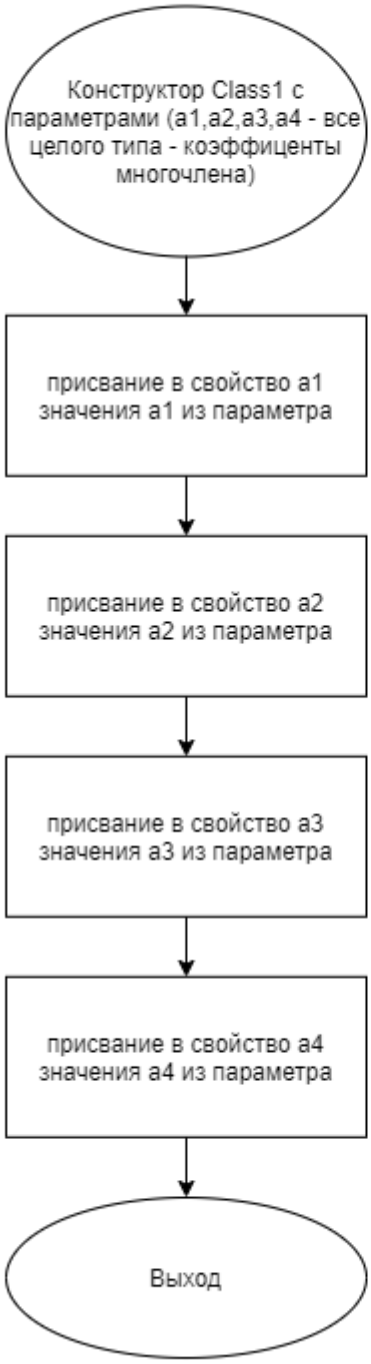
№	Предикат	Действия	№ перехода	Комментарий
1		вывод "\nClass 4"	2	
2		возврат a1*x + a2*x*x + a3*x*x*x + a4*x*x*x*x	∅	выражение степени 4, так как номер класса 4

Блок-схема алгоритма

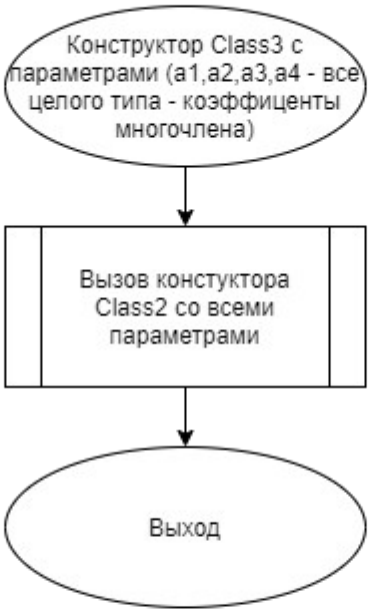

Начало
вызов функции app без параметров
Конец

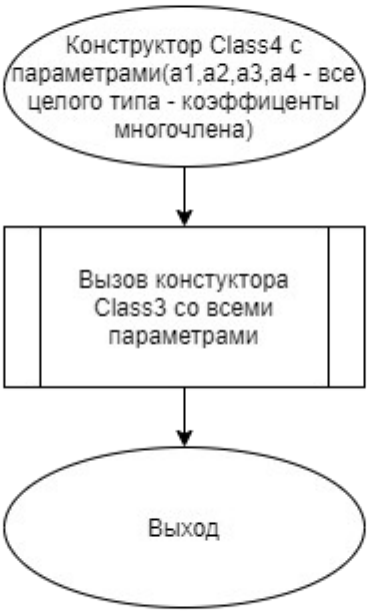

функция app
без параметров
Объявление указателя object
на объект класса Class4
Объявление переменных
целого типа:
a1,a2,a3,a4,x,number,result
ввод a1,a2a,a3,a4
Создание объекта класса Class4 с помощью контструктора,в
который передаются коэффициенты многочлена a1,a2,a3,a4
Вывод коэффицентов многочлена
B
ввод x и номера класса
да
нет
x равен 0
Выход
A

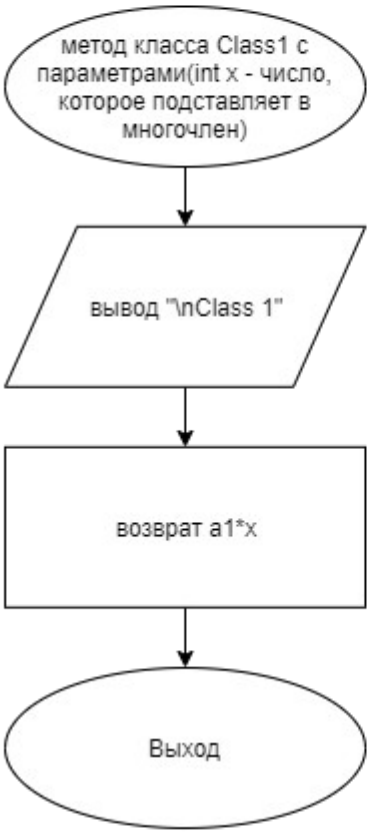

A
Да
number равен 4
Нет
вызов метода get_count с параметром x
Да
number равен 3
Нет
вызов метода Class3::get_count с параметром x
Да
number равен 2
Нет
вызов метода Class2::get_count с параметром x
Да
number равен 1
Нет
вызов метода Class1::get_count с параметром x
присвоение в result результата метода
вывод F(x) = result
B

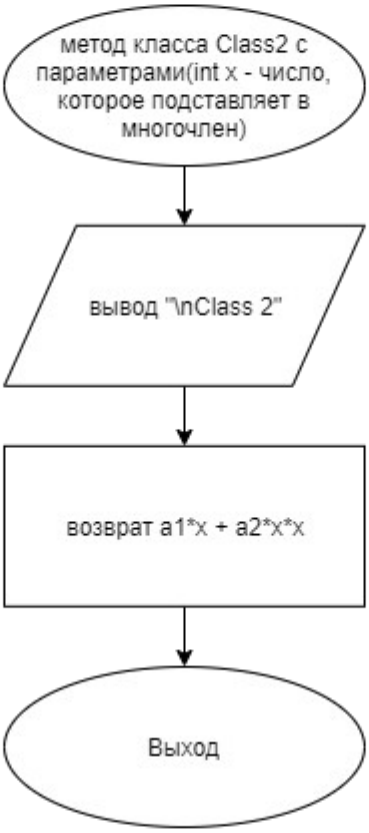

Конструктор Class1 с параметрами (a1,a2,a3,a4 - все целого типа - коэффициенты многочлена)
присвание в свойство a1 значения a1 из параметра
присвание в свойство a2 значения a2 из параметра
присвание в свойство a3 значения a3 из параметра
присвание в свойство a4 значения a4 из параметра
Выход

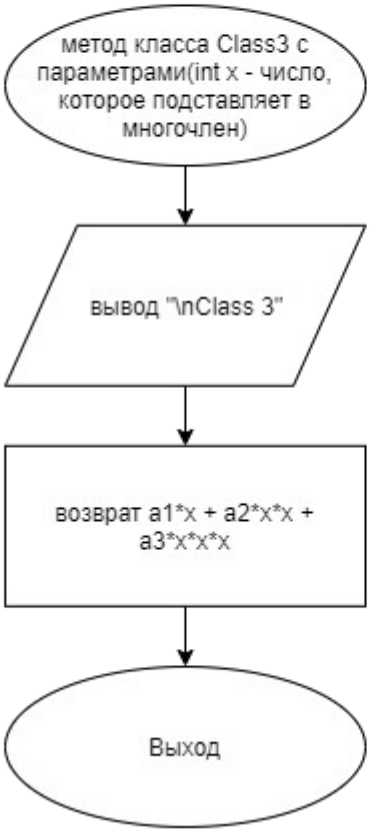

Конструктор Class2 с параметрами(a1,a2,a3,a4 - все целого типа - коэффиценты многочлена)
Вызов констуктора Class1 со всеми параметрами
Выход

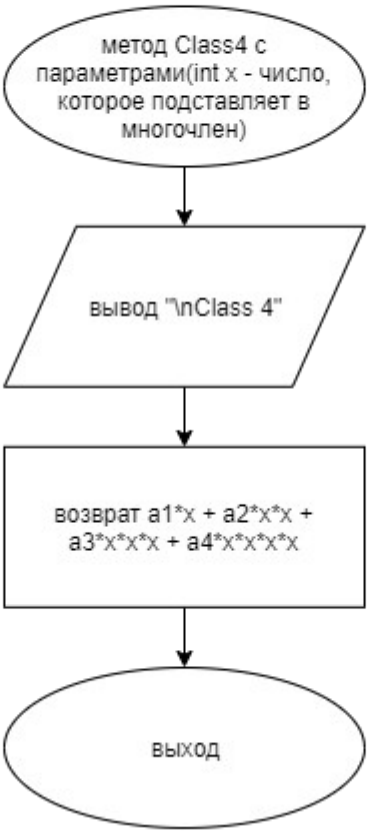

Конструктор Class3 с параметрами (a1,a2,a3,a4 - все целого типа - коэффициенты многочлена)
Вызов констуктора Class2 со всеми параметрами
Выход


Конструктор Class4 с параметрами(a1,a2,a3,a4 - все целого типа - коэффициенты многочлена)
Вызов констуктора Class3 со всеми параметрами
Выход


метод класса Class1 с параметрами(int x - число, которое подставляет в многочлен)
вывод "\nClass 1"
возврат a1*x
Выход


метод класса Class2 с параметрами(int x - число, которое подставляет в многочлен)
вывод "\nClass 2"
возврат a1*x + a2*x*x
Выход


метод класса Class3 с параметрами(int x - число, которое подставляет в многочлен)
вывод "\nClass 3"
возврат a1*x + a2*x*x + a3*x*x*x
Выход



Код программы

Файл app.cpp

```
#include <iostream>
#include "Class4.h"

using namespace std;

void app() {
	Class4* object;
	// объявляю коэффиценты многочлена,x, номер класса и результат
выражения
	int a1,a2,a3,a4,x, number, result;
	cin >> a1 >> a2 >> a3 >> a4;
	object = new Class4(a1,a2,a3,a4);
	// вывод коэффицентов
	cout << "a1 = " << a1 << "    " << "a2 = " << a2 << "    " << "a3 = "
<< a3 << "    " << "a4 = " << a4;
	// начало бесконечного цикла, который закончится при вводе 0
	while(true) {
		// ввод x и номера класса
		cin >> x >> number;
		if(x == 0) break;

		// в зависимости от номерп класса вызываю метода для этого
класса
		if(number == 4) result = object->get_count(x);
		else if(number == 3) result = object->Class3::get_count(x);
		else if(number == 2) result = object->Class2::get_count(x);
		else if(number == 1) result = object->Class1::get_count(x);
		// если номер другой, то ничего не делаю
		else continue;
		// вывожу результат выражения
		cout << "    F( " << x << " ) = " << result;
	}
}
```

Файл app.h

```
#ifndef APP_H
#define APP_H

void app();

#endif;
```

Файл Class1.cpp

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

Class1::Class1(int a1, int a2, int a3, int a4) {
        // присваиваю свойствам класса значения из параметров
        this->a1 = a1;
        this->a2 = a2;
        this->a3 = a3;
        this->a4 = a4;
}

int Class1::get_count(int x) {
        // вывожу информацию, что это 1ый класс
        cout << "\nClass 1";
        // считаю пример 1ой степени
        return a1*x;
}
```

Файл Class1.h

```
#ifndef CLASS_1_H
#define CLASS_1_H
#include <iostream>

class Class1 {
        public:
                Class1(int a1, int a2, int a3, int a4);
                virtual int get_count(int x);
        protected:
                int a1;
                int a2;
                int a3;
                int a4;
};

#endif
```

Файл Class2.cpp

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

// конструктор наследуется
Class2::Class2(int a1, int a2, int a3, int a4): Class1(a1,a2,a3,a4) {}
```

```
int Class2::get_count(int x) {
        // вывожу информацию, что это 2ой класс
        cout << "\nClass 2";
        // считаю пример 2ой степени
        return a1*x + a2*x*x;
}
```

Файл Class2.h

```
#ifndef CLASS_2_H
#define CLASS_2_H
#include <iostream>

#include "Class1.h"

class Class2: public Class1 {
        public:
                Class2(int a1, int a2, int a3, int a4);
                virtual int get_count(int x);
};

#endif
```

Файл Class3.cpp

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

// конструктор наследуется

Class3::Class3(int a1, int a2, int a3, int a4): Class2(a1,a2,a3,a4) {}

int Class3::get_count(int x) {
        // вывожу информацию, что это 3ий класс
        cout << "\nClass 3";
        // считаю пример 3ей степени
        return a1*x + a2*x*x + a3*x*x*x;
}
```

Файл Class3.h

```
#ifndef CLASS_3_H
#define CLASS_3_H
#include <iostream>
#include "Class2.h"

class Class3: public Class2 {
        public:
                Class3(int a1, int a2, int a3, int a4);
                virtual int get_count(int x);
};

#endif
```

Файл Class4.cpp

```
#include "Class3.h"
#include "Class4.h"

#include <iostream>
using namespace std;

// конструктор наследуется
Class4::Class4(int a1, int a2, int a3, int a4): Class3(a1,a2,a3,a4) {}

int Class4::get_count(int x) {
        // вывожу информацию, что это 4ый класс
        cout << "\nClass 4";
        // считаю пример 4ой степени
        return a1*x + a2*x*x + a3*x*x*x + a4*x*x*x*x;
}
```

Файл Class4.h

```
#ifndef CLASS_4_H
#define CLASS_4_H
#include <iostream>
#include "Class3.h"

class Class4: public Class3 {
        public:
                Class4(int a1, int a2, int a3, int a4);
                virtual int get_count(int x);
};
```

```
#endif
```

Файл main.cpp

```
#include <iostream>
#include "app.h"

using namespace std;

int main() {
	// в функции app основной алгоритм
	app();
	return(0);
}
```

Тестирование

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
1 1 1 1 1 4 1 3 1 2 1 1 1 2 0	a1 = 1 a2 = 1 a3 = 1 a4 = 1 Class 4 F(1) = 4 Class 3 F(1) = 3 Class 2 F(1) = 2 Class 1 F(1) = 1 Class 2 F(1) = 2	a1 = 1 a2 = 1 a3 = 1 a4 = 1 Class 4 F(1) = 4 Class 3 F(1) = 3 Class 2 F(1) = 2 Class 1 F(1) = 1 Class 2 F(1) = 2
1 2 3 4 1 4 2 3 3 2 4 4 1 1 0	a1 = 1 a2 = 2 a3 = 3 a4 = 4 Class 4 F(1) = 10 Class 3 F(2) = 34 Class 2 F(3) = 21 Class 4 F(4) = 1252 Class 1 F(1) = 1	a1 = 1 a2 = 2 a3 = 3 a4 = 4 Class 4 F(1) = 10 Class 3 F(2) = 34 Class 2 F(3) = 21 Class 4 F(4) = 1252 Class 1 F(1) = 1