

Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

| | |
|---------------------------------------|----|
| ВВЕДЕНИЕ..... | 4 |
| 1 Постановка задачи..... | 5 |
| 2 Метод решения..... | 8 |
| 3 Описание алгоритма..... | 11 |
| 4 Блок-схема алгоритма..... | 12 |
| 5 Код программы..... | 14 |
| 6 Тестирование..... | 18 |
| ЗАКЛЮЧЕНИЕ..... | 20 |
| СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ..... | 21 |

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 a_1 , a_2 , a_3 , a_4 , значения которых соответствуют коэффициентам многочлена $(a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3 + a_4 \cdot x^4)$.
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 класса 4 посредством параметризованного 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 `ptr` класса `Class4` предназначен для демонстрации работы программы;
- функция `main` для определения входной точки программы;
- библиотека ввода-вывода;
- указатель;
- заголовочные файлы;
- наследование;
- классы.

Класс `Class1`:

- свойства/поля:
 - поле хранит коэффициент полинома:
 - наименование — `ratio`;
 - тип — `int`;
 - модификатор доступа — `private`;
- функционал:
 - метод `Class1` — инициализирует приватное поле `ratio` через параметр;
 - метод `Evaluate` — вычисляет значение полинома.

Класс `Class2`:

- свойства/поля:
 - поле хранит коэффициент полинома:
 - наименование — `ratio`;
 - тип — `int`;
 - модификатор доступа — `private`;

- функционал:
 - метод `Class2` — инициализирует приватное поле `ratio` через параметр;
 - метод `Evaluate` — вычисляет значение полинома.

Класс `Class3`:

- свойства/поля:
 - поле хранит коэффициент полинома:
 - наименование — `ratio`;
 - тип — `int`;
 - модификатор доступа — `private`;
- функционал:
 - метод `Class3` — инициализирует приватное поле `ratio` через параметр;
 - метод `Evaluate` — вычисляет значение полинома.

Класс `Class4`:

- свойства/поля:
 - поле хранит коэффициент полинома:
 - наименование — `ratio`;
 - тип — `int`;
 - модификатор доступа — `private`;
- функционал:
 - метод `Class4` — инициализирует приватное поле `ratio` через параметр;
 - метод `Evaluate` — вычисляет значение полинома.

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 |
| 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 конструктора класса Class1

Функционал: Инициализирует приватное поле ratio через параметр.

Параметры: int num.

Алгоритм конструктора представлен в таблице 2.

Таблица 2 – Алгоритм конструктора класса Class1

| № | Предикат | Действия | № перехода |
|---|----------|---------------------------|---------------|
| 1 | | Инициализация ratio = num | Ø |

3.2 Алгоритм метода Evaluate класса Class1

Функционал: Вычисляет полином.

Параметры: int x.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода Evaluate класса Class1

| № | Предикат | Действия | № перехода |
|---|----------|-----------------------|---------------|
| 1 | | Возврат ratio * x * x | Ø |

3.3 Алгоритм конструктора класса Class2

Функционал: Инициализирует приватное поле ratio через параметр.

Параметры: int num, int num2.

Алгоритм конструктора представлен в таблице 4.

Таблица 4 – Алгоритм конструктора класса Class2

| № | Предикат | Действия | № перехода |
|---|----------|----------------------------|---------------|
| 1 | | Инициализация ratio = num2 | Ø |

3.4 Алгоритм метода Evaluate класса Class2

Функционал: Вычисляет значение полинома.

Параметры: int x.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода Evaluate класса Class2

| № | Предикат | Действия | № перехода |
|---|----------|---|---------------|
| 1 | | Возврат ratio * x * x + Class1::Evaluate(x) | Ø |

3.5 Алгоритм конструктора класса Class3

Функционал: Инициализирует приватное поле ratio через параметр.

Параметры: int num, int num2, int num3.

Алгоритм конструктора представлен в таблице 6.

Таблица 6 – Алгоритм конструктора класса Class3

| № | Предикат | Действия | № перехода |
|---|----------|----------------------------|---------------|
| 1 | | Инициализация ratio = num3 | Ø |

3.6 Алгоритм метода Evaluate класса Class3

Функционал: Вычисляет значение полинома.

Параметры: int x.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода Evaluate класса Class3

| № | Предикат | Действия | № перехода |
|---|----------|---|---------------|
| 1 | | Возврат ratio * x * x * x + Class2::Evaluate(x) | Ø |

3.7 Алгоритм конструктора класса Class4

Функционал: Инициализирует приватное поле ratio через параметр.

Параметры: int num, int num2, int num3, int num4.

Алгоритм конструктора представлен в таблице 8.

Таблица 8 – Алгоритм конструктора класса Class4

| № | Предикат | Действия | № перехода |
|---|----------|----------------------------|---------------|
| 1 | | Инициализация ratio = num4 | Ø |

3.8 Алгоритм метода Evaluate класса Class4

Функционал: Вычисляет значение полинома.

Параметры: int x.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 9.

Таблица 9 – Алгоритм метода Evaluate класса Class4

| № | Предикат | Действия | № перехода |
|---|----------|---|---------------|
| 1 | | Возврат ratio * x * x * x * x + Class3::Evaluate(x) | Ø |

3.9 Алгоритм функции main

Функционал: Входная точка программы.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм функции представлен в таблице 10.

Таблица 10 – Алгоритм функции main

| № | Предикат | Действия | № перехода |
|----|-------------|--|---------------|
| 1 | | Объявление указателя на объект класса Class4 | 2 |
| 2 | | Объявление целочисленных переменных a1, a2, a3, a4 | 3 |
| 3 | | Объявление целочисленной переменной x | 4 |
| 4 | | Объявление целочисленной переменной i_class | 5 |
| 5 | | Ввод a1, a2, a3, a4 | 6 |
| 6 | | Инициализация указателя ptr на объект класса Class4 с помощью оператора new с передачей a1, a2, a3, a4 в качестве параметров | 7 |
| 7 | | Вывод a1 = a1 a2 = a2 a3 = a3 a4 = a4 | 8 |
| 8 | пока true | Ввод x | 9 |
| | | | 14 |
| 9 | x == 0 | | 14 |
| | | | 10 |
| 10 | | Ввод i_class | 11 |
| 11 | | Вывод "Class " i_class " F(x) = " | 12 |
| 12 | i_class = 1 | Вывод ptr->Class1::Evaluate(x) | 13 |

| № | Предикат | Действия | № перехода |
|----|-------------|---|---------------|
| | i_class = 2 | Вывод ptr->Class2::Evaluate(x) | 13 |
| | i_class = 3 | Вывод ptr->Class3::Evaluate(x) | 13 |
| | i_class = 4 | Вывод ptr->Class4::Evaluate(x) | 13 |
| 13 | | Вывод переноса строки | 8 |
| 14 | | Удаление объекта ptr с помощью оператора delete с освобождением памяти указателя | ∅ |

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-5.

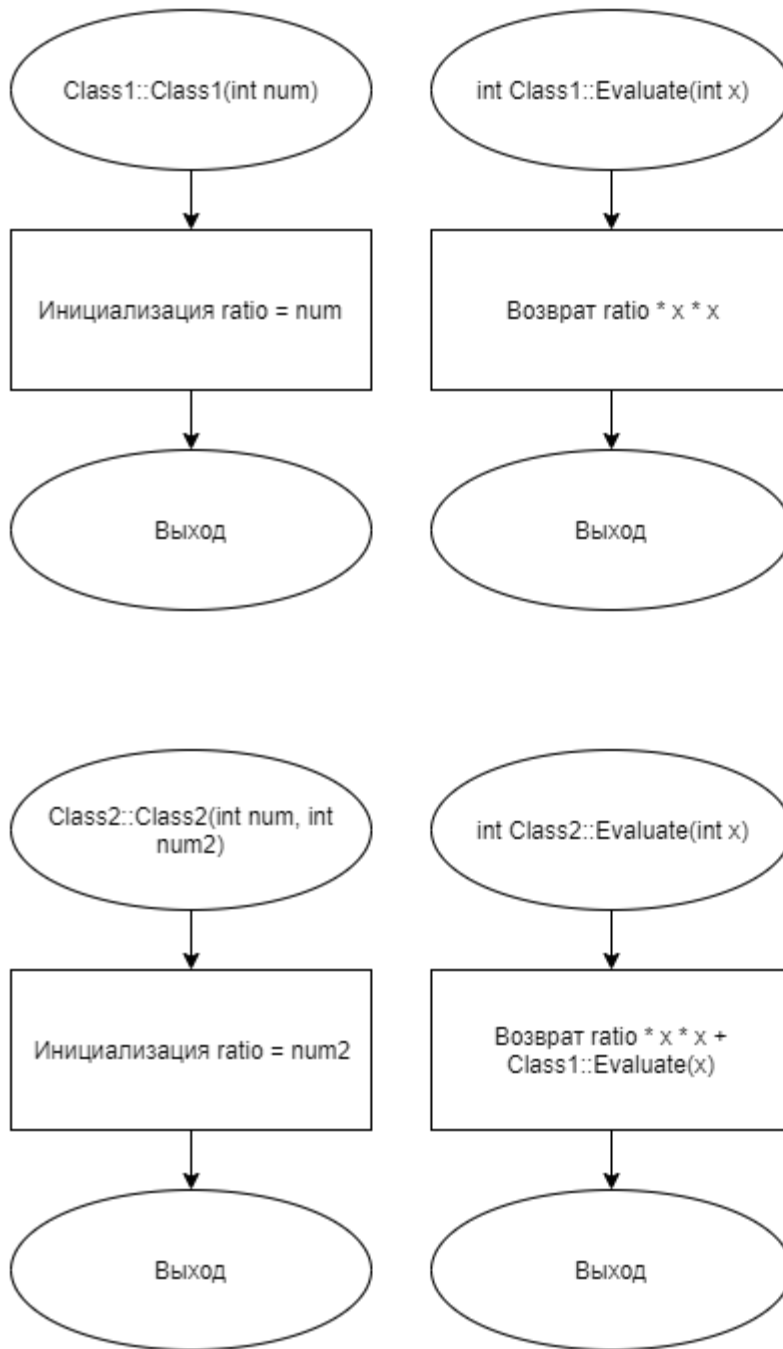


Рисунок 1 – Блок-схема алгоритма

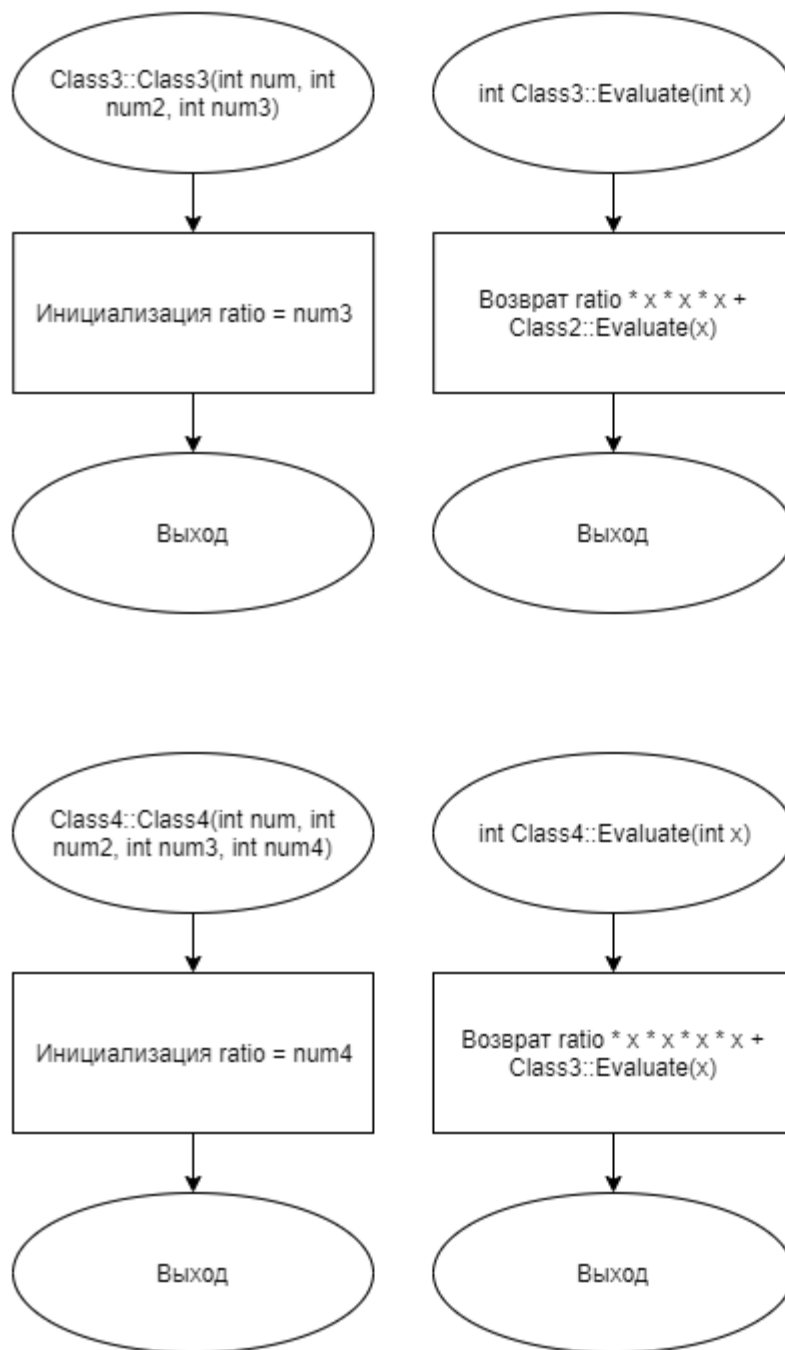


Рисунок 2 – Блок-схема алгоритма

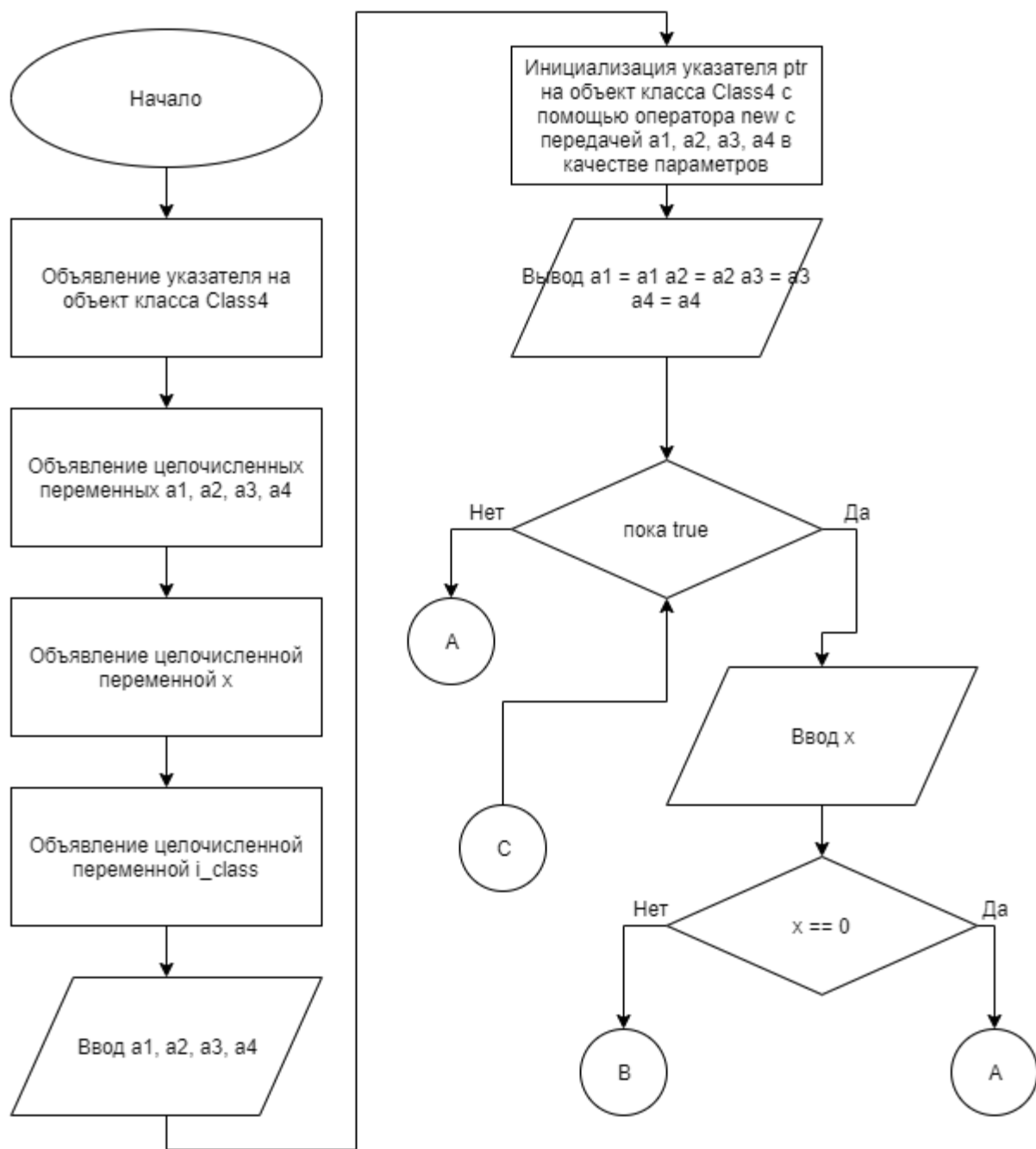


Рисунок 3 – Блок-схема алгоритма

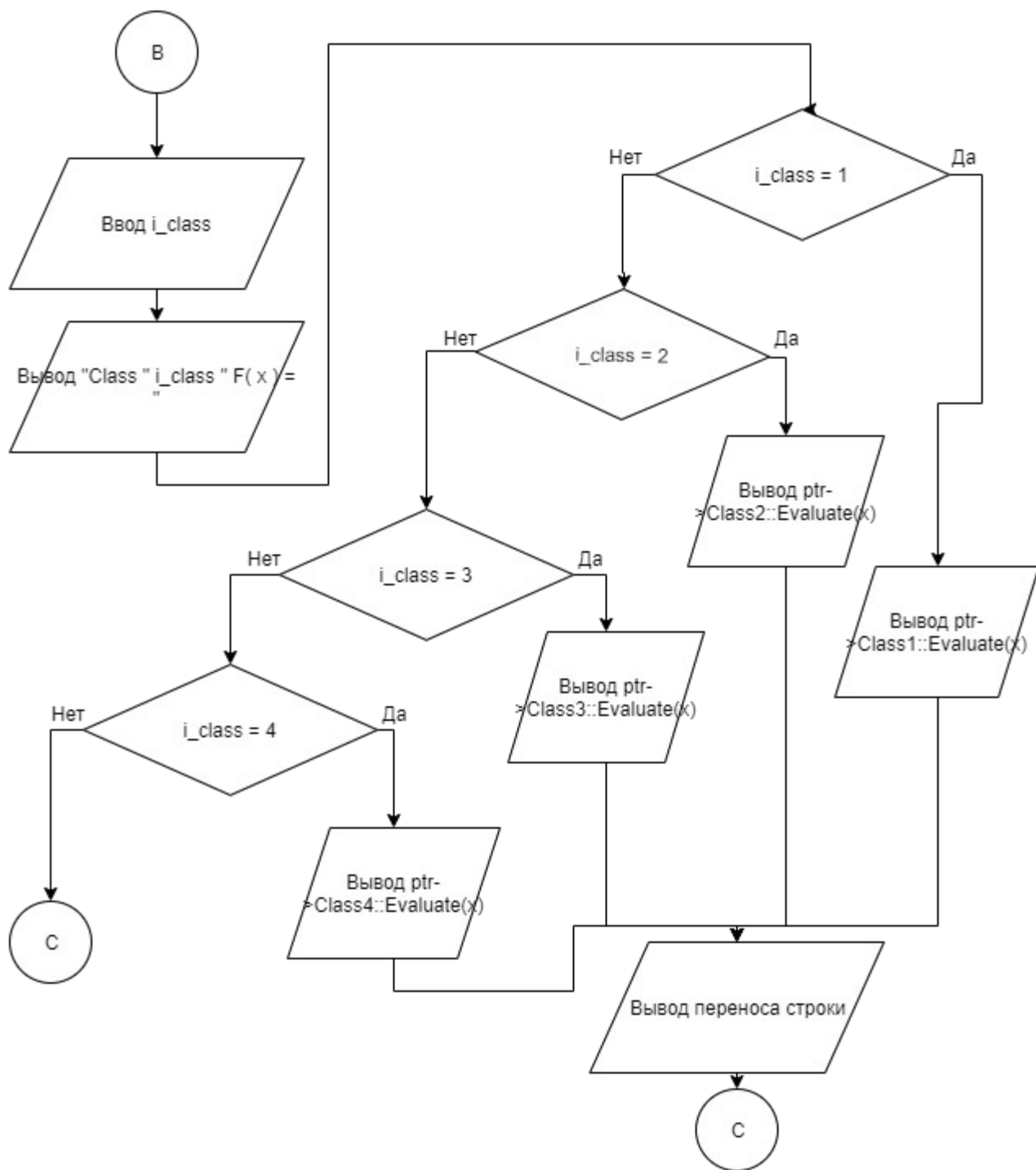


Рисунок 4 – Блок-схема алгоритма

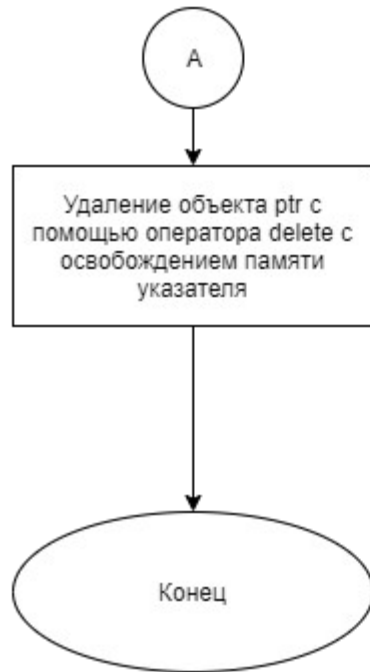


Рисунок 5 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл Class1.cpp

Листинг 1 – Class1.cpp

```
#include "Class1.h"

Class1::Class1(int num)
{
    ratio = num;
}

int Class1::Evaluate(int x)
{
    return ratio * x;
}
```

5.2 Файл Class1.h

Листинг 2 – Class1.h

```
#ifndef __CLASS1__H
#define __CLASS1__H

class Class1 {
    int ratio;
public:
    Class1(int num);
    int Evaluate(int x);
};

#endif
```

5.3 Файл Class2.cpp

Листинг 3 – Class2.cpp

```
#include "Class2.h"

Class2::Class2(int num, int num2) : Class1(num)
{
    ratio = num2;
}

int Class2::Evaluate(int x)
{
    return ratio * x * x + Class1::Evaluate(x);
}
```

5.4 Файл Class2.h

Листинг 4 – Class2.h

```
#ifndef __CLASS2__H
#define __CLASS2__H

#include "Class1.h"

class Class2 : public Class1 {
    int ratio;
public:
    Class2(int num, int num2);
    int Evaluate(int x);
};

#endif
```

5.5 Файл Class3.cpp

Листинг 5 – Class3.cpp

```
#include "Class3.h"

Class3::Class3(int num, int num2, int num3) : Class2(num, num2)
{
    ratio = num3;
}
```

```

}

int Class3::Evaluate(int x)
{
    return ratio * x * x * x + Class2::Evaluate(x);
}

```

5.6 Файл Class3.h

Листинг 6 – Class3.h

```

#ifndef __CLASS3__H
#define __CLASS3__H

#include "Class2.h"

class Class3 : public Class2 {
    int ratio;
public:
    Class3(int num, int num2, int num3);
    int Evaluate(int x);
};

#endif

```

5.7 Файл Class4.cpp

Листинг 7 – Class4.cpp

```

#include "Class4.h"

Class4::Class4(int num, int num2, int num3, int num4) : Class3(num, num2,
num3)
{
    ratio = num4;
}

int Class4::Evaluate(int x)
{
    return ratio * x * x * x * x + Class3::Evaluate(x);
}

```

5.8 Файл Class4.h

Листинг 8 – Class4.h

```
#ifndef __CLASS4__H
#define __CLASS4__H

#include "Class3.h"

class Class4 : public Class3 {
    int ratio;
public:
    Class4(int num, int num2, int num3, int num4);
    int Evaluate(int x);
};

#endif
```

5.9 Файл main.cpp

Листинг 9 – main.cpp

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

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

    std::cin >> a1 >> a2 >> a3 >> a4;

    ptr = new Class4(a1, a2, a3, a4);

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

    while (true)
    {
        std::cin >> x;
        if (x == 0) break;
        std::cin >> i_class;

        std::cout << "Class " << i_class << "    " << "F( " << x << " )" << " =
";
        if (i_class == 1)
```

```
        {
            std::cout << ptr->Class1::Evaluate(x);
        }
        else if (i_class == 2)
        {
            std::cout << ptr->Class2::Evaluate(x);
        }
        else if (i_class == 3)
        {
            std::cout << ptr->Class3::Evaluate(x);
        }
        else if (i_class == 4)
        {
            std::cout << ptr->Class4::Evaluate(x);
        }
        std::cout << std::endl;
    }

    delete ptr;

    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 2 3 4 2 1 2 2 2 3 2 4 0 | a1 = 1 a2 = 2 a3 = 3 a4 = 4 Class 1 F(2) = 2 Class 2 F(2) = 10 Class 3 F(2) = 34 Class 4 F(2) = 98 | a1 = 1 a2 = 2 a3 = 3 a4 = 4 Class 1 F(2) = 2 Class 2 F(2) = 10 Class 3 F(2) = 34 Class 4 F(2) = 98 |

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-avvora.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-avvora.ru/student/files/Prilozheniye_k_methodichke.pdf (дата обращения 05.05.2021).
4. Шилдт Г. С++: 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).