Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ.....6
1.1 Описание входных данных.....7
1.2 Описание выходных данных.....7
2 МЕТОД РЕШЕНИЯ.....8
3 ОПИСАНИЕ АЛГОРИТМОВ.....10
3.1 Алгоритм конструктора класса cl_1.....10
3.2 Алгоритм метода resolve класса cl_1.....10
3.3 Алгоритм конструктора класса cl_2.....10
3.4 Алгоритм метода resolve класса cl_2.....11
3.5 Алгоритм конструктора класса cl_3.....11
3.6 Алгоритм метода resolve класса cl_3.....12
3.7 Алгоритм конструктора класса cl_4.....12
3.8 Алгоритм метода resolve класса cl_4.....12
3.9 Алгоритм функции main.....13
4 БЛОК-СХЕМЫ АЛГОРИТМОВ.....14
5 КОД ПРОГРАММЫ.....19
5.1 Файл cl_1.cpp.....19
5.2 Файл cl_1.h.....19
5.3 Файл cl_2.cpp.....20
5.4 Файл cl_2.h.....20
5.5 Файл cl_3.cpp.....20
5.6 Файл cl_3.h.....21
5.7 Файл cl_4.cpp.....21
5.8 Файл cl_4.h.....22
5.9 Файл main.cpp.....22
6 ТЕСТИРОВАНИЕ.....24

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....25

1 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
    - 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.

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:

- функция main для Основная функция;
- Объект стандартного потока ввода/ вывода cin/cout.

Класс cl_1:

- свойства/поля:
    - поле Значение переменной:
        - наименование — int_1;
        - тип — Целочисленное;
        - модификатор доступа — protected;
- функционал:
    - метод cl_1 — Параметризированный конструктор;
    - метод resolve — Виртуальный метод вычисления значения.

Класс cl_2:

- свойства/поля:
    - поле Значение переменной:
        - наименование — int_2;
        - тип — Целочисленное;
        - модификатор доступа — protected;
- функционал:
    - метод cl_2 — Параметризированный конструктор;
    - метод resolve — Виртуальный метод вычисления значения.

Класс cl_3:

- свойства/поля:
    - поле Значение переменной:
        - наименование — int_3;

        - тип — Целочисленное;
        - модификатор доступа — protected;
- функционал:
    - метод cl_3 — Параметризированный конструктор;
    - метод resolve — Виртуальный метод вычисления значения.

Класс cl_4:

- свойства/поля:
    - поле Значение переменной:
        - наименование — int_4;
        - тип — Целочисленное;
        - модификатор доступа — protected;
- функционал:
    - метод cl_4 — Параметризированный конструктор;
    - метод resolve — Виртуальный метод вычисления значения.

Таблица 1 – Иерархия наследования классов

№	Имя класса	Классы-наследники	Модификатор доступа при наследовании	Описание	Номер
1	cl_1				
		cl_2	public		2
2	cl_2				
		cl_3	public		3
3	cl_3				
		cl_4	public		4
4	cl_4				

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 cl_1

Функционал: Параметризированный конструктор.

Параметры: нет.

Алгоритм конструктора представлен в таблице 2.

Таблица 2 – Алгоритм конструктора класса cl_1

№	Предикат	Действия	№ перехода
1		Параметризированный конструктор	∅

3.2 Алгоритм метода resolve класса cl_1

Функционал: Виртуальный метод вычисления значения.

Параметры: нет.

Возвращаемое значение: Целочисленное.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода resolve класса cl_1

№	Предикат	Действия	№ перехода
1		Возвращение значения int_1 * x	∅

3.3 Алгоритм конструктора класса cl_2

Функционал: Параметризированный конструктор.

Параметры: нет.

Алгоритм конструктора представлен в таблице 4.

Таблица 4 – Алгоритм конструктора класса cl_2

№	Предикат	Действия	№ перехода
1		Параметризированный конструктор	∅

3.4 Алгоритм метода resolve класса cl_2

Функционал: Виртуальный метод вычисления значения.

Параметры: нет.

Возвращаемое значение: Целочисленное.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода resolve класса cl_2

№	Предикат	Действия	№ перехода
1		Возвращение значения int_2 * x * x	∅

3.5 Алгоритм конструктора класса cl_3

Функционал: Параметризированный конструктор.

Параметры: нет.

Алгоритм конструктора представлен в таблице 6.

Таблица 6 – Алгоритм конструктора класса cl_3

№	Предикат	Действия	№ перехода
1		Параметризированный конструктор	∅

3.6 Алгоритм метода resolve класса cl_3

Функционал: Виртуальный метод вычисления значения.

Параметры: нет.

Возвращаемое значение: Целочисленное.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода resolve класса cl_3

№	Предикат	Действия	№ перехода
1		Возвращение значения int_3 * x * x * x	∅

3.7 Алгоритм конструктора класса cl_4

Функционал: Параметризированный конструктор.

Параметры: нет.

Алгоритм конструктора представлен в таблице 8.

Таблица 8 – Алгоритм конструктора класса cl_4

№	Предикат	Действия	№ перехода
1		Параметризированный конструктор	∅

3.8 Алгоритм метода resolve класса cl_4

Функционал: Виртуальный метод вычисления значения.

Параметры: нет.

Возвращаемое значение: Целочисленное.

Алгоритм метода представлен в таблице 9.

Таблица 9 – Алгоритм метода resolve класса cl_4

№	Предикат	Действия	№ перехода
1		Возвращение значения int_4 * x * x * x * x	∅

3.9 Алгоритм функции main

Функционал: Основная функция.

Параметры: нет.

Возвращаемое значение: Целочилсенное.

Алгоритм функции представлен в таблице 10.

Таблица 10 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		Объявление указателя на объект класса	2
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
3		Объявление целочисленной переменной x, которая соответствует переменной многочлена	4
4		Ввод значения переменных a1, a2, a3 a4	5
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
6		Выполнение цикла	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

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

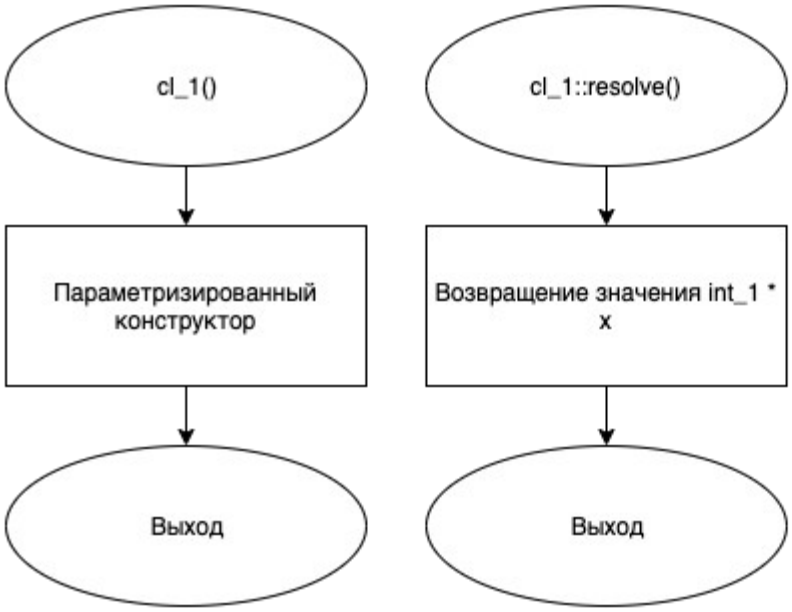

Рисунок 1 – Блок-схема алгоритма

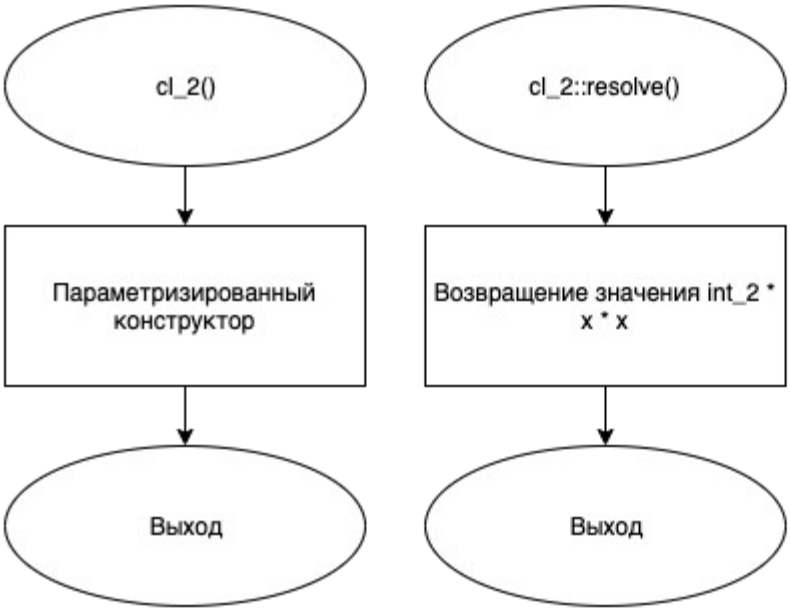

Рисунок 2 – Блок-схема алгоритма

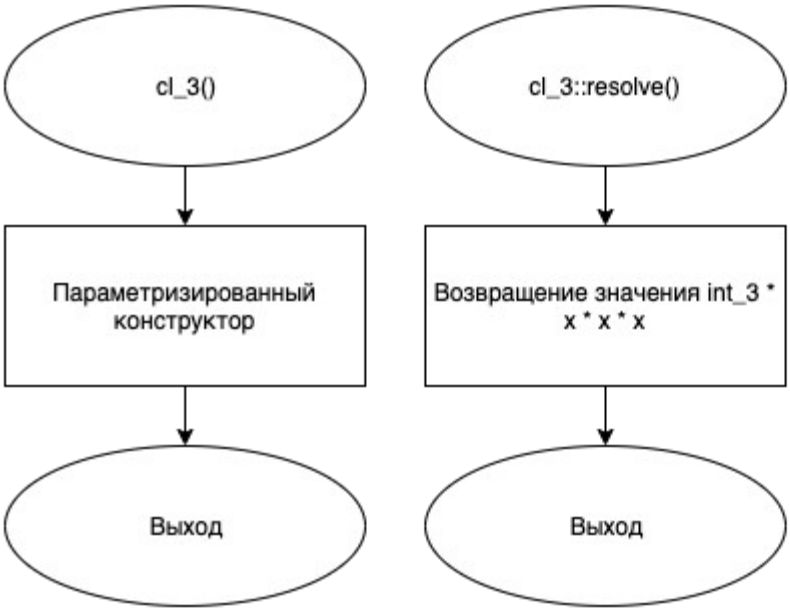

Рисунок 3 – Блок-схема алгоритма

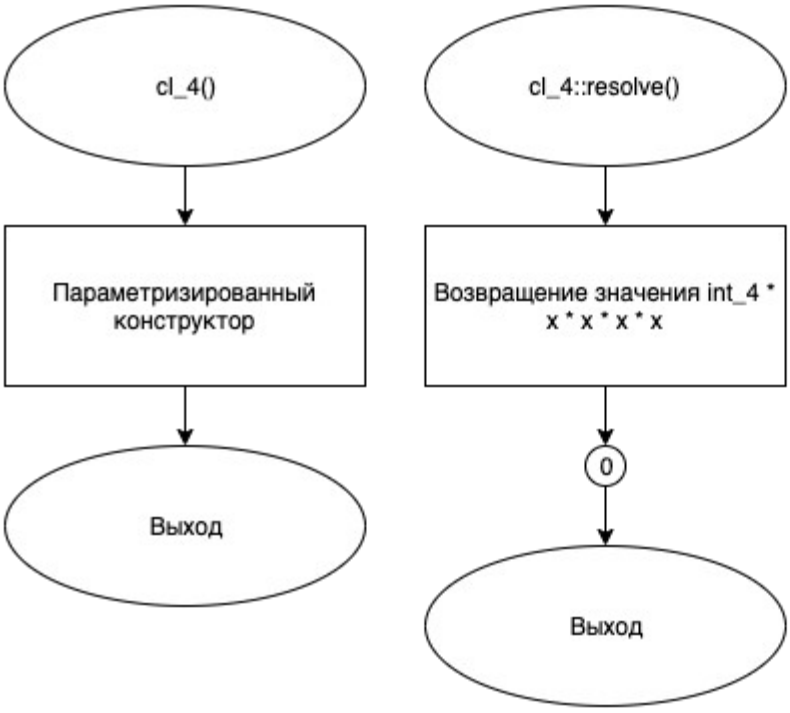

Рисунок 4 – Блок-схема алгоритма

Рисунок 5 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл cl_1.cpp

Листинг 1 – cl_1.cpp

```
#include "cl_1.h"

cl_1::cl_1(int int_1, int int_2, int int_3, int int_4)\
{
   this->int_1 = int_1;
   this->int_2 = int_2;
   this->int_3 = int_3;
   this->int_4 = int_4;
}

int cl_1::resolve(int x)
{
   return int_1 * x;
}
```

5.2 Файл cl_1.h

Листинг 2 – cl_1.h

```
#ifndef __CL_1__H
#define __CL_1__H

#include <iostream>
class cl_1
{
public:
   cl_1(int int_1, int int_2, int int_3, int int_4);
   virtual int resolve(int);
protected:
   int int_1;
   int int_2;
   int int_3;
   int int_4;
};
```

```
#endif
```

5.3 Файл cl_2.cpp

Листинг 3 – cl_2.cpp

```
#include "cl_2.h"

cl_2::cl_2(int int_1, int int_2, int int_3, int int_4):cl_1(int_1,int_2,
int_3, int_4){}

int cl_2::resolve(int x)
{
   return int_1 * x + int_2 * x * x;
}
```

5.4 Файл cl_2.h

Листинг 4 – cl_2.h

```
#ifndef __CL_2__H
#define __CL_2__H

#include "cl_1.h"

class cl_2: public cl_1
{
public:
   cl_2(int int_1, int int_2, int int_3, int int_4);
   virtual int resolve(int);
};

#endif
```

5.5 Файл cl_3.cpp

Листинг 5 – cl_3.cpp

```
#include "cl_3.h"
```

```
cl_3::cl_3(int int_1, int int_2, int int_3, int int_4):cl_2(int_1, int_2,
int_3, int_4){}

int cl_3::resolve(int x)
{
   return int_1 * x + int_2 * x * x + int_3 * x * x * x;
}
```

5.6 Файл cl_3.h

Листинг 6 – cl_3.h

```
#ifndef __CL_3__H
#define __CL_3__H

#include "cl_2.h"

class cl_3: public cl_2
{
public:
   cl_3(int int_1, int int_2, int int_3, int int_4);
   virtual int resolve(int);
};

#endif
```

5.7 Файл cl_4.cpp

Листинг 7 – cl_4.cpp

```
#include "cl_4.h"

cl_4::cl_4(int int_1, int int_2, int int_3, int int_4):cl_3(int_1, int_2,
int_3, int_4){}

int cl_4::resolve(int x)
{
   return int_1 * x + int_2 * x * x + int_3 * x * x * x + int_4 * x * x * x *
x;
}
```

5.8 Файл cl_4.h

Листинг 8 – cl_4.h

```
#ifndef __CL_4__H
#define __CL_4__H

#include "cl_3.h"

class cl_4: public cl_3
{
public:
   cl_4(int int_1, int int_2, int int_3, int int_4);
   virtual int resolve(int);
};

#endif
```

5.9 Файл main.cpp

Листинг 9 – main.cpp

```
#include <stdlib.h>
#include <stdio.h>
#include <iostream>
#include "cl_4.h"
using namespace std;
int main()
{
   cl_4* obj;
   int int_1, int_2, int_3, int_4, x, id, result;
   cin >> int_1 >> int_2 >> int_3 >> int_4;
   obj = new cl_4(int_1, int_2, int_3, int_4);

   cout << "a1 = " << int_1 << "    a2 = " << int_2 << "    a3 = " << int_3
<< "    a4 = " << int_4;
   while(true)
   {
      cin >> x;
      if(x == 0)
      {
         break;
      }
      cin >> id;
      if(id < 1 || id > 4)
      {
         continue;
      }
```

```
            switch(id)
            {
               case 1:
                  result = obj ->cl_1::resolve(x);
                  break;
               case 2:
                  result = obj->cl_2::resolve(x);
                  break;
               case 3:
                  result = obj-> cl_3::resolve(x);
                  break;
               case 4:
                  result = obj->resolve(x);
                  break;
               default:
                  result = 0;
                  break;
            }
            cout << endl << "Class " << id << "    F( "<< x << " ) = " << result;
         }
         return(0);
      }
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 11.

Таблица 11 – Результат тестирования программы

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
4 3 2 1 9 1 8 2 7 3 6 4 0	a1 = 4 a2 = 3 a3 = 2 a4 = 1 Class 1 F(9) = 36 Class 2 F(8) = 224 Class 3 F(7) = 861 Class 4 F(6) = 1860	a1 = 4 a2 = 3 a3 = 2 a4 = 1 Class 1 F(9) = 36 Class 2 F(8) = 224 Class 3 F(7) = 861 Class 4 F(6) = 1860

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).