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 **первого 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.

При сдаче задания пояснить правила использования указателя на объект родительского класса.

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

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 функции main

Функционал: Входная точка программы.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм функции представлен в таблице 1.

Таблица 1 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		Объявление указателя ptr на объект класса Class1	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 классаClass4 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

№	Предикат	Действия	№ перехода
11		Вывод "Class " i_class " F(x) = "	12
12	i_class = 1	Вывод ((Class1*)(ptr))->Evaluate(x)	13
	i_class = 2	Вывод ((Class2*)(ptr))->Evaluate(x)	13
	i_class = 3	Вывод ((Class3*)(ptr))->Evaluate(x)	13
	i_class = 4	Вывод ((Class4*)(ptr))->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 1-3.

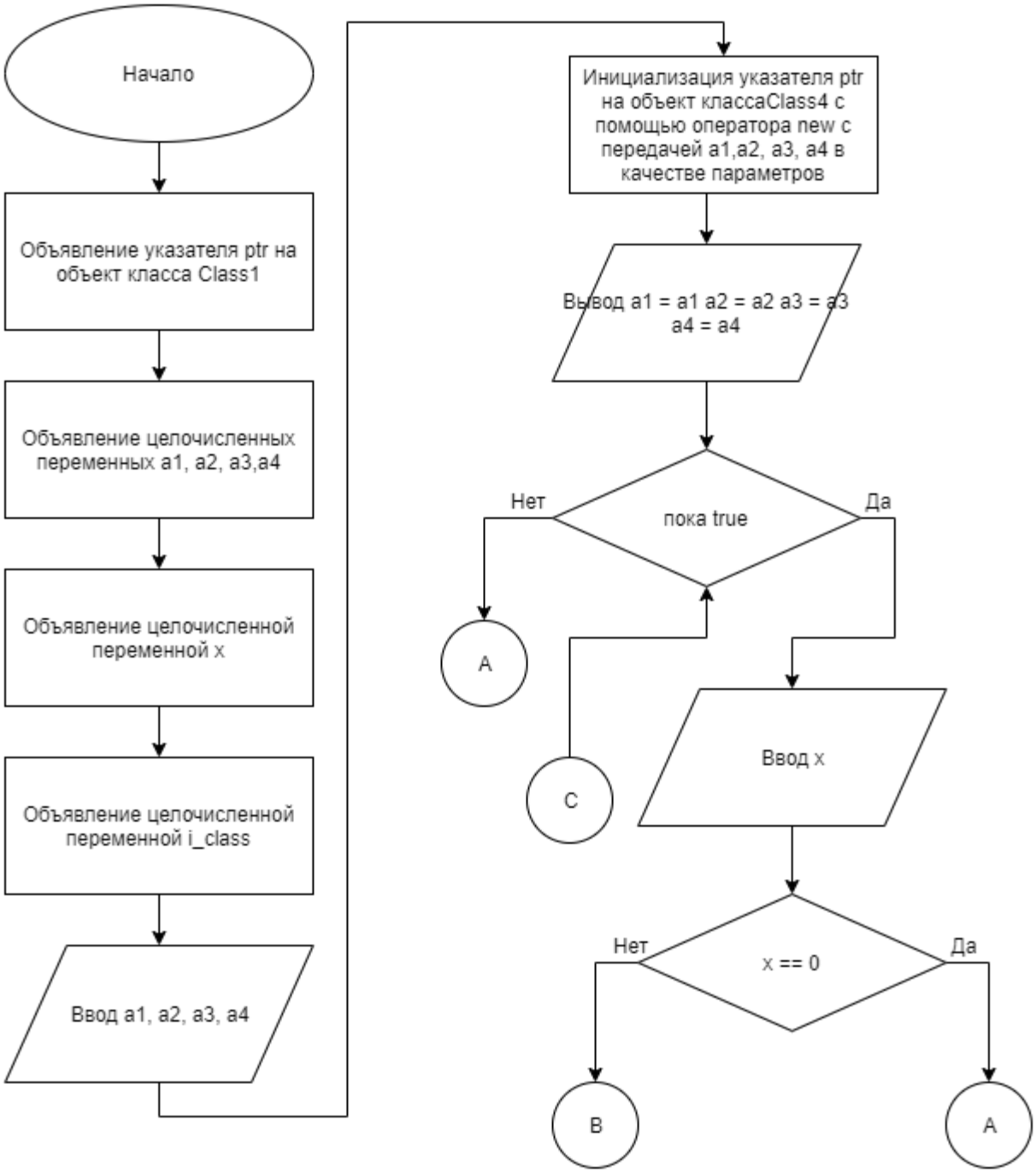

Рисунок 1 – Блок-схема алгоритма

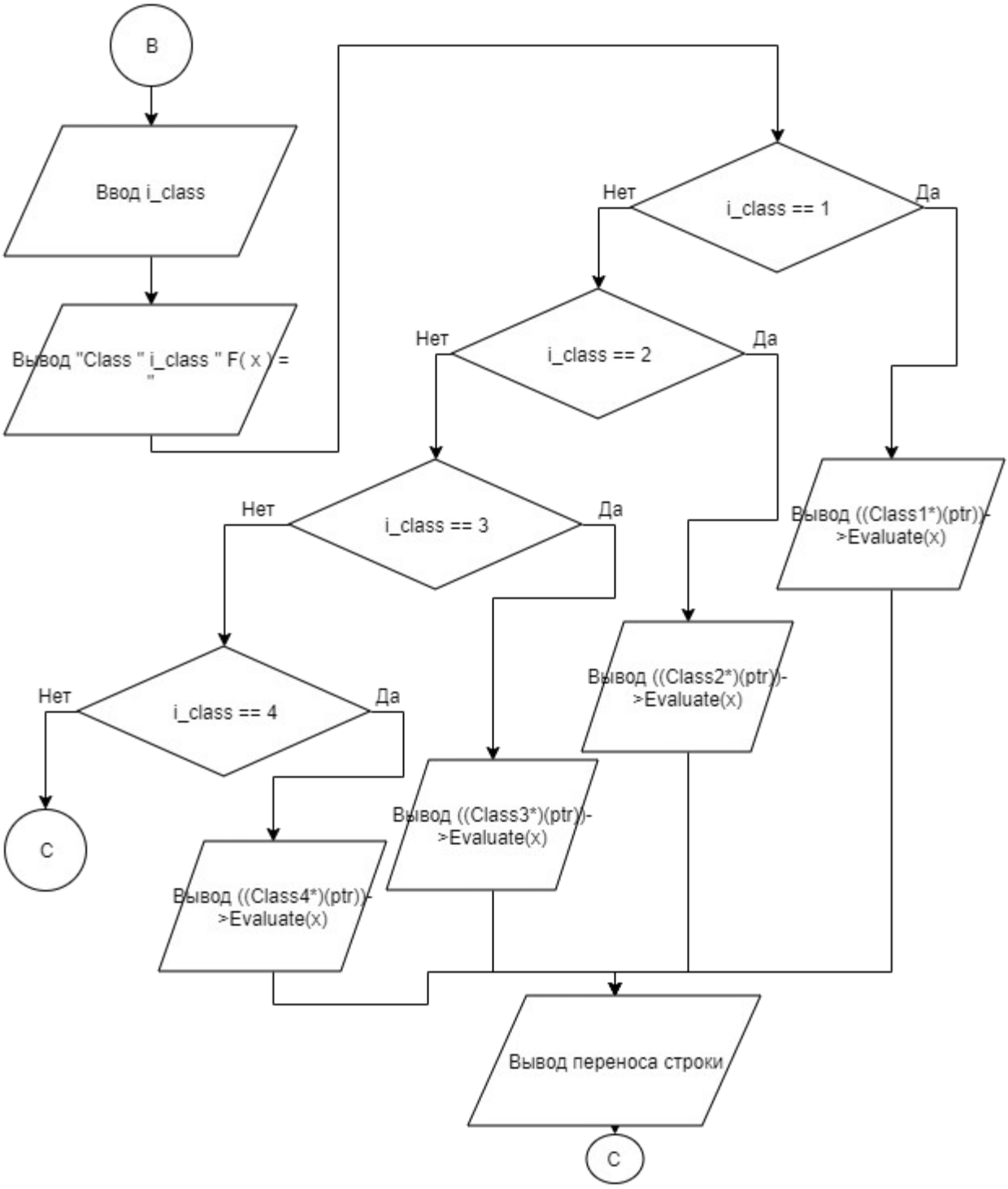

Рисунок 2 – Блок-схема алгоритма

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

         delete ptr;

         return 0;
      }
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 2.

Таблица 2 – 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-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).