Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ.....5
1.1 Описание входных данных.....5
1.2 Описание выходных данных.....6
2 МЕТОД РЕШЕНИЯ.....7
3 ОПИСАНИЕ АЛГОРИТМОВ.....9
3.1 Алгоритм конструктора класса type_1.....9
3.2 Алгоритм метода bind класса type_2.....9
3.3 Алгоритм функции func.....10
3.4 Алгоритм метода func класса type_1.....10
3.5 Алгоритм метода func класса type_2.....11
3.6 Алгоритм функции main.....11
4 БЛОК-СХЕМЫ АЛГОРИТМОВ.....13
5 КОД ПРОГРАММЫ.....15
5.1 Файл main.cpp.....15
5.2 Файл type_1.cpp.....15
5.3 Файл type_1.h.....16
5.4 Файл type_2.cpp.....16
5.5 Файл type_2.h.....17
6 ТЕСТИРОВАНИЕ.....18
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....19

1 ПОСТАНОВКА ЗАДАЧИ

Создать объект первого типа, у которого одно целочисленное свойство. Значение данного свойства определяется посредством параметризированного конструктора.

Создать объект второго типа, у которого две целочисленных свойства. Значение данных свойств определяется посредством метода объекта.

Реализовать дружественную функцию, которая находит максимальное значение полей объекта первого типа и полей объекта второго типа.

Написать программу:

1. Вводит значение для поля объекта первого типа.

2. Создает объект первого типа.

3. Вводит значения полей для полей объекта второго типа.

4. Создает объект второго типа.

5. Определяет значения полей объекта второго типа.

6. Определяет максимальное значение полей, созданных двух объектов разного типа посредством дружественной функции.

7. Выводит полученный результат.

1.1 Описание входных данных

Первая строка:

«целое число в десятичном формате»

Вторая строка:

«целое число в десятичном формате»␣«целое число в десятичном формате»

1.2 Описание выходных данных

Первая строка, с первой позиции:

max = «целочисленное значение в десятеричном формате»

2 МЕТОД РЕШЕНИЯ

Для решения задачи используется:

- объект object_1 класса type_1 предназначен для объект первого типа;
- объект object_2 класса type_2 предназначен для объект второго типа;
- функция main для основная функция программы;
- функция func для функция для нахождения максимума из полей объектов;
- cin - объект стандартного потока ввода с клавиатуры;
- cout - объект стандартного потока вывода на экран;
- if .. else - условный оператор.

Класс type_1:

- свойства/поля:
    - поле целочисленное значение объекта:
        - наименование — value;
        - тип — int;
        - модификатор доступа — private;
- функционал:
    - метод type_1 — параметризированный конструктор;
    - метод func — дружественная функция.

Класс type_2:

- свойства/поля:
    - поле первое целочисленное значение объекта:
        - наименование — value_1;
        - тип — int;
        - модификатор доступа — private;
    - поле второе целочисленное значение объекта:
        - наименование — value_2;

        - тип — int;
        - модификатор доступа — private;
- функционал:
    - метод bind — метод привязки значений параметров полям объекта;
    - метод func — дружественная функция.

Таблица 1 – Иерархия наследования классов

№	Имя класса	Классы-наследники	Модификатор доступа при наследовании	Описание	Номер
1	type_1			класс превого типа	
2	type_2			класс второго типа	

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 type_1

Функционал: параметризированный конструктор.

Параметры: int value.

Алгоритм конструктора представлен в таблице 2.

Таблица 2 – Алгоритм конструктора класса type_1

№	Предикат	Действия	№ перехода
1		присваивание полю объекта value значение параметра value	∅

3.2 Алгоритм метода bind класса type_2

Функционал: метод привязки значений параметров полям объекта.

Параметры: int value_1, int value_2.

Возвращаемое значение: void.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода bind класса type_2

№	Предикат	Действия	№ перехода
1		присваивание полю value_1 значение параметра value_1	2
2		присваивание полю value_2 значение параметра value_2	∅

3.3 Алгоритм функции func

Функционал: дружественная функция.

Параметры: type_1 object_1, type_2 object_2.

Возвращаемое значение: int.

Алгоритм функции представлен в таблице 4.

Таблица 4 – Алгоритм функции func

№	Предикат	Действия	№ перехода
1		инициализация переменных целочисленных переменных a значением поля value объекта object_1, b значением поля value_1 объекта object_2 и c значением поля value_2 объекта object_2	2
2	a > b && a > c	возврат значения переменной a	∅
	b > a && b > c	возврат значения переменной b	∅
		возврат значения переменной c	∅

3.4 Алгоритм метода func класса type_1

Функционал: дружественная функция.

Параметры: type_1 object_1, type_2 object_2.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода func класса type_1

№	Предикат	Действия	№ перехода
1			∅

3.5 Алгоритм метода func класса type_2

Функционал: дружественная функция.

Параметры: type_1 object_1, type_2 object_2.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 6.

Таблица 6 – Алгоритм метода func класса type_2

№	Предикат	Действия	№ перехода
1			∅

3.6 Алгоритм функции main

Функционал: основная функция программы.

Параметры: нет.

Возвращаемое значение: int - код ошибки.

Алгоритм функции представлен в таблице 7.

Таблица 7 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		создание целочисленных переменных value_1, value_2	2
2		ввод значения переменной value_1 с клавиатуры	3
3		создание объекта object_1 класса type_1 с помощью параметризированного конструктора с аргументом value_1	4
4		ввод значений переменных value_1 и value_2 с клавитуры	5
5		создание объекта object_2 класса type_2	6
6		вызов метода bind объекта object_2 с параметрами value_1, value_2	7
7		инициализация целочисленной переменной max значением результата функции func с параметрами object_1, object_2	8
8		вывод на экран "max = " и значение переменной max	9

№	Предикат	Действия	№ перехода
9		возврат значения 0	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-2.

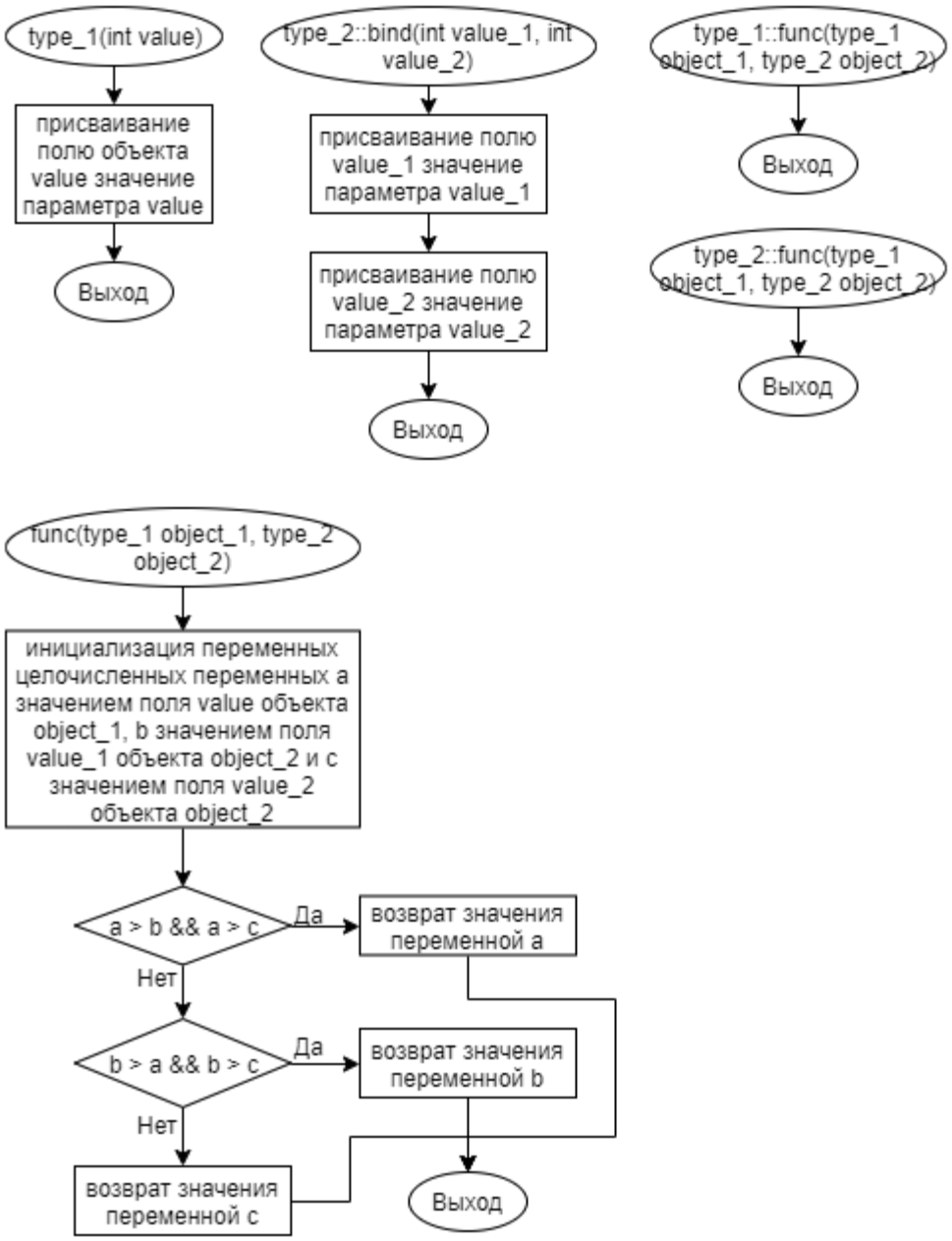

Рисунок 1 – Блок-схема алгоритма

Рисунок 2 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл main.cpp

Листинг 1 – main.cpp

```
#include <stdlib.h>
#include <stdio.h>
#include <iostream>
#include "type_1.h"
#include "type_2.h"

int func(type_1 object_1, type_2 object_2)
{
   int a = object_1.value, b = object_2.value_1, c = object_2.value_2;
   if (a > b && a > c)
      return a;
   else if (b > a && b > c)
      return b;
   else
      return c;
}

int main()
{
   int value_1, value_2;
   std::cin >> value_1;
   type_1 object_1(value_1);
   std::cin >> value_1 >> value_2;
   type_2 object_2;
   object_2.bind(value_1, value_2);
   int max = func(object_1, object_2);
   std::cout << "max = " << max;
   return(0);
}
```

5.2 Файл type_1.cpp

Листинг 2 – type_1.cpp

```
#include "type_1.h"
```

```
#include "type_2.h"

type_1::type_1(int value)
{
   this -> value = value;
}
```

5.3 Файл type_1.h

Листинг 3 – type_1.h

```
#ifndef __TYPE_1__H
#define __TYPE_1__H
#include "type_2.h"

class type_2;
class type_1
{
private:
   int value;
public:
   type_1(int value);
   friend int func(type_1 object_1, type_2 object_2);
};

#endif
```

5.4 Файл type_2.cpp

Листинг 4 – type_2.cpp

```
#include "type_2.h"

void type_2::bind(int value_1, int value_2)
{
   this -> value_1 = value_1;
   this -> value_2 = value_2;
}
```

5.5 Файл type_2.h

Листинг 5 – type_2.h

```
#ifndef __TYPE_2__H
#define __TYPE_2__H
#include "type_1.h"

class type_1;
class type_2
{
private:
   int value_1;
   int value_2;
public:
   void bind(int value_1, int value_2);
   friend int func(type_1 object_1, type_2 object_2);
};

#endif
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 8.

Таблица 8 – Результат тестирования программы

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
5 3 2	max = 5	max = 5
1 7 8	max = 8	max = 8
4 9 3	max = 9	max = 9

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).