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 1) демонстрирует возможность конструирования производного объекта на базе исходного объекта; 2) выполняет перераспределение прав доступа к элементам исходного объекта; 3) демонстрирует механизм однозначного обращения (использования, доступа) к элементам производного и исходного объекта.

Спроектировать исходный объект (разработать описание класса), который имеет элементы:

В закрытом доступе:

- одно свойство целого типа;
- метод, с одним целочисленным параметром, который меняет значение свойства в закрытом доступе на утроенное значение параметра.

В открытом доступе:

- одно свойство целого типа;
- параметризированный конструктор, с двумя целочисленными параметрами, который устанавливает значения свойств с закрытым и открытым доступом. Значение закрытого свойства меняется посредством вызова метода из закрытого раздела;
- метод с двумя целочисленными параметрами, который устанавливает значения свойств с закрытым и открытым доступом. Значение закрытого свойства меняется посредством вызова метода из закрытого доступа;
- метод, который выводит на экран значение обоих свойств. Сперва значение закрытого свойства, потом значение открытого свойства.

Спроектировать производный объект (разработать описание класса производного объекта), который содержит исходный объект и имеет элементы:

В закрытом доступе:

- одно свойство целого типа, наименование которого совпадает с наименование закрытого свойства исходного объекта;

В открытом доступе:

- одно свойство целого типа, наименование которого совпадает с наименование открытого свойства исходного объекта;
- параметризированный конструктор, с двумя целочисленными параметрами, который устанавливает значения свойств с закрытым и открытым доступом;
- метод с двумя целочисленными параметрами, который устанавливает значения свойств с закрытым и открытым доступом. Наименование метода совпадает с наименованием аналогичного метода исходного объекта;
- метод, который выводит на экран значение обоих свойств. Сперва значение свойства закрытым доступом, потом значение свойства открытым доступом. Наименование метода совпадает с наименованием аналогичного метода исходного объекта.

Производный объект спроектировать таким образом, чтобы к открытым элементам исходного объекта сохранить открытый доступ.

Алгоритм конструирования и отработки системы:

1. Добавление в состав системы двух целочисленных переменных.

2. Ввод значений двух целочисленных переменных.

3. Объявление объекта производного класса. При этом обеспечивается отработка параметризированного конструктора для исходного объекта и для производного. В качестве аргументов используются целочисленные переменные, в последовательности, как им были присвоены значения. Первый аргумент содержит значение для закрытого свойства, второй - для открытого свойства.

4. Вывод значений свойств исходного объекта.

5. Вывод значений свойств производного объекта.

6. Ввод значения одной целочисленной переменной.

7. Присвоение введенного значения открытому свойству производного объекта.

8. Присвоение введенного значения открытому свойству исходного объекта.

9. Вывод значений свойств производного объекта.

10. Вывод значений свойств родительского объекта

11. Ввод значений двух целочисленных переменных.

12. Если значение первой переменной больше нуля, то:

12.1. Переопределение значений свойств производного объекта - увеличение на единицу введенных исходных значений.

12.2. Переопределение значений свойств исходного объекта - уменьшение на единицу введенных исходных значений.

12.3. Вывод значений свойств производного объекта.

12.4. Вывод значений свойств родительского объекта.

13. Иначе:

13.1. Переопределение значений свойств родительского объекта - увеличение на единицу введенных исходных значений.

13.2. Переопределение значений свойств производного объекта - уменьшение на единицу введенных исходных значений.

13.3. Вывод значений свойств родительского объекта.

13.4. Вывод значений свойств производного объекта.

14. Завершение работы системы.

1.1 Описание входных данных

В первой строке:

«Целое число» «Целое число»

Во второй строке:

```
«Целое число»
```

В третьей строке:

```
«Целое число» «Целое число»
```

Пример ввода

```
8 5
11
-7 12
```

1.2 Описание выходных данных

Начиная с первой строки:

```
«Целое число»    «Целое число»
«Целое число»    «Целое число»
«Целое число»    «Целое число»
«Целое число»    «Целое число»
«Целое число»    «Целое число»
«Целое число»    «Целое число»
```

Пример вывода

```
24    5
8    5
8    11
24    11
-18    13
-8    11
```

2 МЕТОД РЕШЕНИЯ

Для решения задачи используется:

- объект drv класса Child предназначен для демонстрации возможности конструирования производного объекта на базе исходного объекта;
- библиотека iostream;
- оператор присваивания =;
- арифметические операторы сложения и вычитания +, -;
- объекты стандартного потока ввода/вывода данных cin, cout;
- операторы сравнения <, >;
- оператор ветвления if ... else.

Класс Parent:

- свойства/поля:
    - поле закрытое целочисленное свойство:
        - наименование — locked_num;
        - тип — int;
        - модификатор доступа — private;
    - поле открытое целочисленное свойство:
        - наименование — num;
        - тип — int;
        - модификатор доступа — public;
- функционал:
    - метод Parent — параметризированный конструктор;
    - метод triple_locked — метод, изменяющий закрытое свойство locked_num;
    - метод pout — вывод значений всех полей класса;
    - метод change — изменение всех полей класса.

Класс Child:

- свойства/поля:
  - поле закрытое целочисленное свойство:
    - наименование — locked_num;
    - тип — int;
    - модификатор доступа — private;
  - поле открытое целочисленное свойство:
    - наименование — num;
    - тип — int;
    - модификатор доступа — public;
- функционал:
  - метод Child — параметризированный конструктор;
  - метод pout — вывод значений всех полей класса;
  - метод change — изменение всех полей класса.

Таблица 1 – Иерархия наследования классов

№	Имя класса	Классы-наследники	Модификатор доступа при наследовании	Описание	Номер
1	Parent				
		Child	public		2
2	Child				

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 Parent

Функционал: параметризированный конструктор.

Параметры: num1 - значение для изменения закрытого свойства, num2 - значение для изменения открытого свойства.

Алгоритм конструктора представлен в таблице 2.

Таблица 2 – Алгоритм конструктора класса Parent

№	Предикат	Действия	№ перехода
1		вызов метода triple_locked с параметром num1	2
2		присвоение значению num значения num2	∅

3.2 Алгоритм метода triple_locked класса Parent

Функционал: метод, изменяющий закрытое свойство locked_num.

Параметры: нет.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода triple_locked класса Parent

№	Предикат	Действия	№ перехода
1		присвоение значению поля locked_num утроенного значения nm	∅

3.3 Алгоритм метода pout класса Parent

Функционал: вывод значений всех полей класса.

Параметры: нет.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода pout класса Parent

№	Предикат	Действия	№ перехода
1		вывод закрытого поля locked_num	2
2		вывод 4 пробелов и открытого поля num	3
3		переход на новую строку	∅

3.4 Алгоритм метода change класса Parent

Функционал: изменение всех полей класса.

Параметры: num1 - значение для изменения закрытого свойства, num2 - значение для изменения открытого свойства.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода change класса Parent

№	Предикат	Действия	№ перехода
1		вызов метода triple_locked с параметром num1	2
2		присвоение значения num2 значению поля num	∅

3.5 Алгоритм конструктора класса Child

Функционал: параметризированный конструктор.

Параметры: num1 - значение для изменения закрытого свойства, num2 -

значение для изменения открытого свойства.

Алгоритм конструктора представлен в таблице 6.

Таблица 6 – Алгоритм конструктора класса Child

№	Предикат	Действия	№ перехода
1		вызов метода triple_locked с параметром num1	2
2		присвоение значения num2 значению поля num	∅

3.6 Алгоритм метода pout класса Child

Функционал: вывод значений всех полей класса.

Параметры: нет.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода pout класса Child

№	Предикат	Действия	№ перехода
1		вывод закрытого поля locked_num	2
2		вывод 4 пробелов и открытого поля num	3
3		переход на новую строку	∅

3.7 Алгоритм метода change класса Child

Функционал: изменение всех полей класса.

Параметры: num1 - значение для изменения закрытого свойства, num2 - значение для изменения открытого свойства.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 8.

Таблица 8 – Алгоритм метода change класса Child

№	Предикат	Действия	№ перехода
1		присвоение значения num1 значению закрытого поля locked_num	2
2		присвоение значения num2 значению открытого поля num	∅

3.8 Алгоритм функции main

Функционал: главная функция программы.

Параметры: нет.

Возвращаемое значение: целое, идентификатор работоспособности программы.

Алгоритм функции представлен в таблице 9.

Таблица 9 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		объявление целочисленных переменных num1, num2	2
2		ввод значений для целочисленных переменных num1, num2	3
3		объявление объекта drv класса Child. При этом обеспечивается отработка параметризированного конструктора для исходного объекта и для производного. В качестве аргументов используются целочисленные переменные num1 и num2	4
4		вызов метода родительского pout для объекта drv	5
5		вызов метода pout для объекта drv	6
6		ввод значения num2 значению родительского поля объекта drv	7
7		присвоение значения num2 значению	8

№	Предикат	Действия	№ перехода
		родительского поля объекта drv	
8		присвоение значения num2 значению поля объекта drv	9
9		вызов метода pout для объекта drv	10
10		вызов метода родительского pout для объекта drv	11
11		ввод значений для целочисленных переменных num1 и num2	12
12	num1 > 0?	вызов метода change с параметрами num1+1, num2+1	13
		вызов метода change с параметрами num1-1, num2-1	16
13		вызов родительского метода change с параметрами num1-1, num2-1	14
14		вызов метода pout для объекта drv	15
15		вызов метода родительского pout для объекта drv	∅
16		вызов родительского метода change с параметрами num1+1, num2+1	17
17		вызов метода родительского pout для объекта drv	18
18		вызов метода pout для объекта drv	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-4.

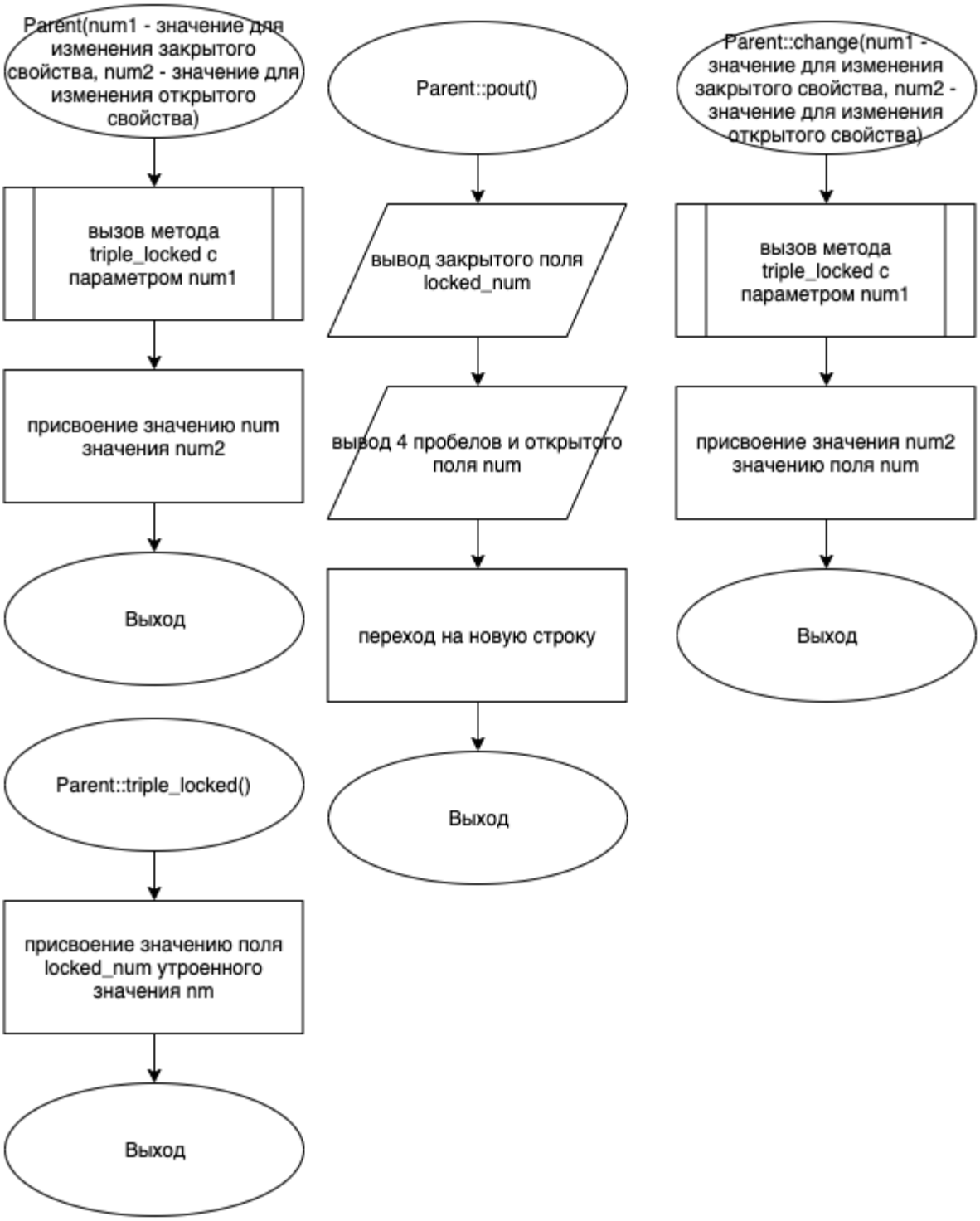

Рисунок 1 – Блок-схема алгоритма

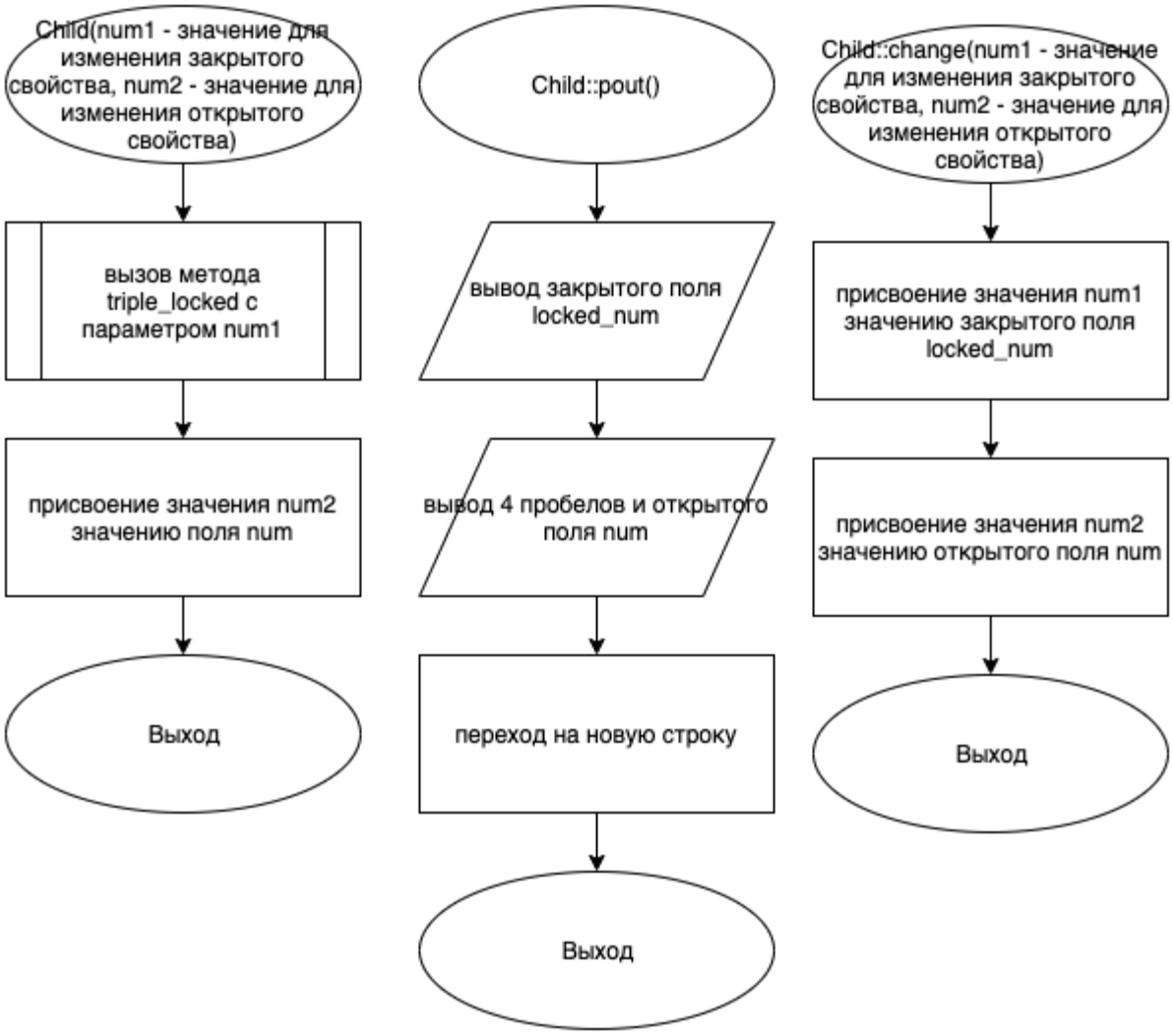

Рисунок 2 – Блок-схема алгоритма

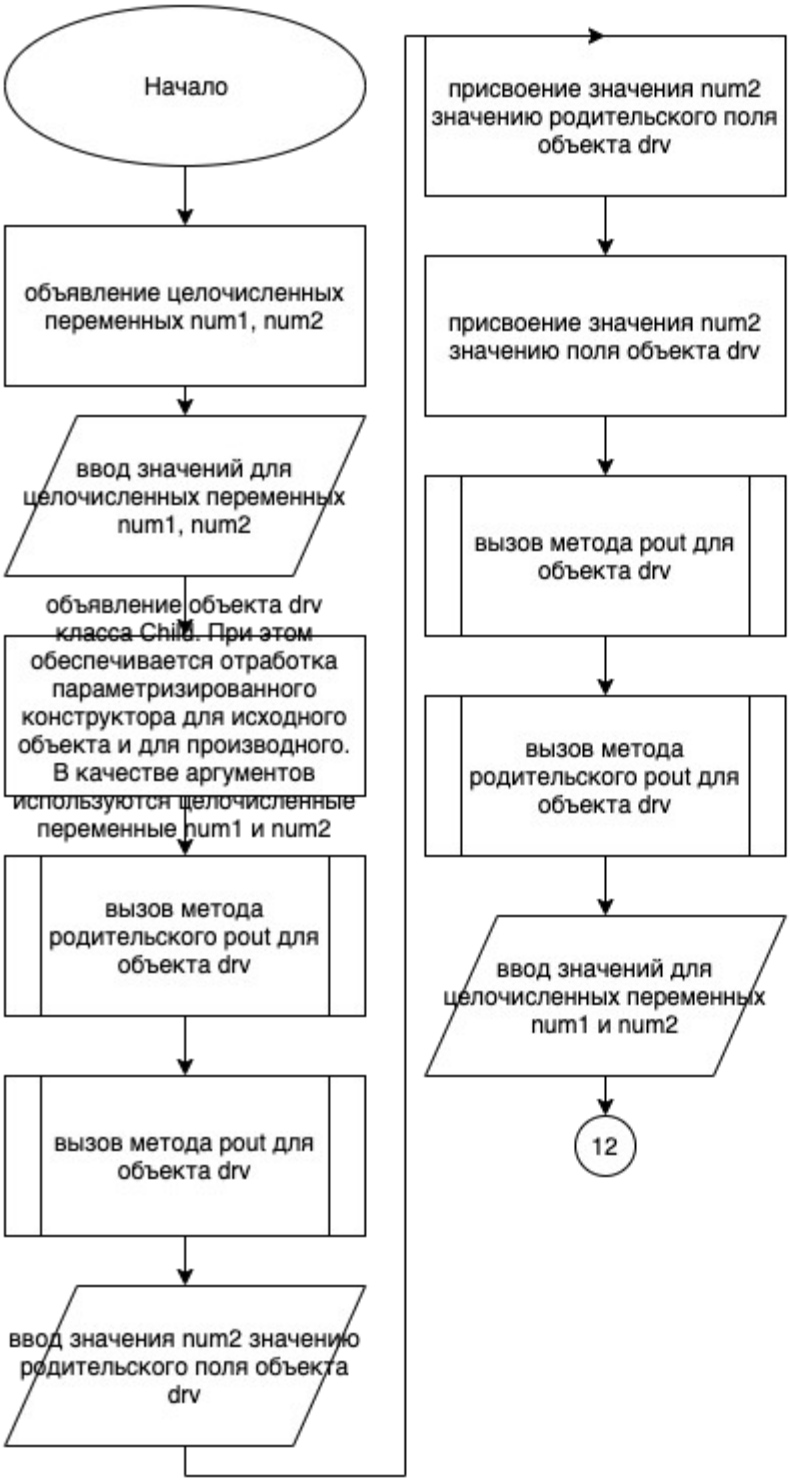

Рисунок 3 – Блок-схема алгоритма

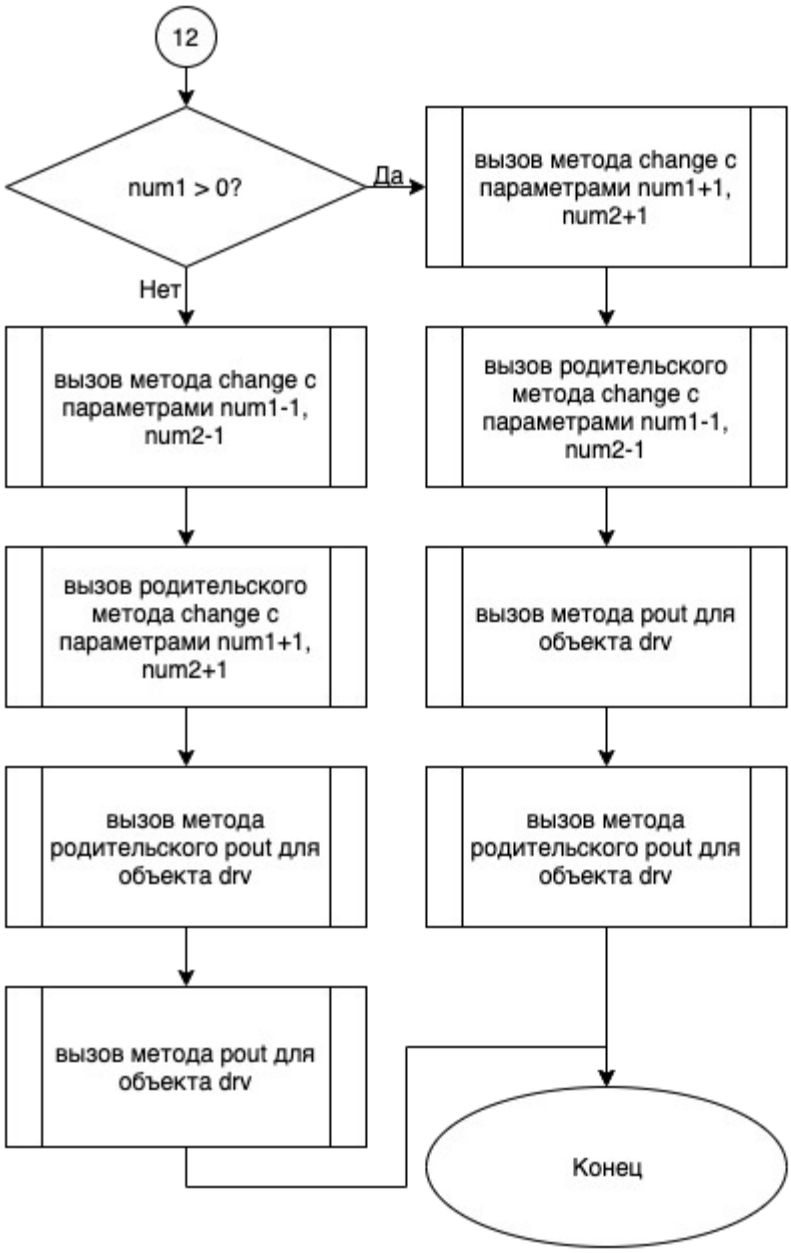

Рисунок 4 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл Child.cpp

Листинг 1 – Child.cpp

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

Child::Child(int num1, int num2) : Parent(num1, num2){
   locked_num = num1;
   num = num2;
}

void Child::change(int num1, int num2){
   locked_num = num1;
   num = num2;
}

void Child::pout(){
   cout << locked_num << "    " << num << endl;
}
```

5.2 Файл Child.h

Листинг 2 – Child.h

```
#ifndef __CHILD__H
#define __CHILD__H
#include "Parent.h"

class Child : public Parent {
   int locked_num;
   public:
      int num;
      Child (int num1, int num2);
      void change (int num1, int num2);
      void pout();
};
```

```
#endif
```

5.3 Файл main.cpp

Листинг 3 – main.cpp

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

int main()
{
   int num1, num2;
   cin >> num1 >> num2;

   Child drv(num1, num2);
   drv.Parent::pout();
   drv.pout();

   cin >> num2;

   drv.Parent::num = num2;
   drv.num = num2;
   drv.pout();
   drv.Parent::pout();

   cin >> num1 >> num2;

   if(num1 > 0){
      drv.change(num1 + 1, num2 + 1);
      drv.Parent::change(num1 - 1, num2 - 1);
      drv.pout();
      drv.Parent::pout();
   } else {
      drv.change(num1 - 1, num2 - 1);
      drv.Parent::change(num1 + 1, num2 + 1);
      drv.Parent::pout();
      drv.pout();
   }
   return(0);
}
```

5.4 Файл Parent.cpp

Листинг 4 – Parent.cpp

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

Parent::Parent(int num1, int num2){
   triple_locked(num1);
   num = num2;
}

void Parent::triple_locked(int nm){
   locked_num = 3 * nm;
}

void Parent::change(int num1, int num2){
   triple_locked(num1);
   num = num2;
}

void Parent::pout(){
   cout << locked_num << "    " << num << endl;
}
```

5.5 Файл Parent.h

Листинг 5 – Parent.h

```
#ifndef __PARENT__H
#define __PARENT__H

class Parent{
   int locked_num;
   void triple_locked(int nm);
   public:
      int num;
      Parent(int num1, int num2);
      void change(int num1, int num2);
      void pout();
};

#endif
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 10.

Таблица 10 – Результат тестирования программы

Входные данные	**Ожидаемые выходные данные**	**Фактические выходные данные**
8 5 11 -7 12	24 5 8 5 8 11 24 11 -18 13 -8 11	24 5 8 5 8 11 24 11 -18 13 -8 11

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).