Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

Разработать систему, которая демонстрирует возможность использования объекта дружественного класса.

Система состоит из множества однотипных объектов. Объекты содержат характеристику, на которую влияет внешнее возмущение, что приводит к отклонению части значений за пределы допустимой нормы. В составе системы входит инструмент, который периодически анализирует состояние каждого объекта и при обнаружении недопустимых отклонений проводит необходимые корректировки, для восстановления режима нормального функционирования объектов.

Спроектировать объект, с свойствами в закрытом доступе:

- целого типа, для хранения размерности массива;
- целого типа, для хранения величины допустимого отклонения;
- указатель на объект целого типа.

С параметризированным конструктором. У конструктора есть параметр целого типа. Параметр передает (содержит) значение размерности целочисленного массива. У конструктора есть параметр целого типа. Параметр передает (содержит) значение величины допустимого отклонения. В конструкторе: фиксируются значения размерности массива и величины допустимого отклонения; создается целочисленный массив заданной размерности; вводится значения элементов массива. Класс данного объекта является родительским. Имеется функционал (методы) в открытом доступе:

- реализует возмущение значений элементов целочисленного массива. Метод имеет одни целочисленный параметр, который содержит величину возмущения для элементов массива. Каждому элементу массива добавляется величина возмущения;

- реализует вывод значений элементов массива.

В деструкторе освобождается память, выделенная для массива.

Спроектировать производный объект, на базе родительского объекта. У объекта в закрытом доступе имеется свойство:

- строкового типа, для хранения наименования объекта.

С параметризированным конструктором. У конструктора есть параметр целого типа. Параметр передает (содержит) значение размерности целочисленного массива. У конструктора есть параметр целого типа. Параметр передает (содержит) значение величины допустимого отклонения. В реализации конструктора вводится значение наименования объекта.

Имеется функционал (метод) в открытом доступе, который с новой строки выводит наименование объекта и значения элементов массива.

Спроектировать объект, который анализирует состояние производных объектов. Проверяет значения всех элементов массива. Если значение выходит за рамки интервала допустимости, то значение элемента меняет на ноль. Для этого у объекта имеется соответствующий метод. Допустимым интервалом является отрицательное и положительного значение величины отклонения. Например: [-5, 5].

Алгоритм конструирования и отработки системы:

1. Объявляются целочисленные переменные для хранения значений: количества объектов; размерности массива; величины допустимого отклонения; количество итераций.

2. Объявляется строковая переменная, для хранения наименования объекта.

3. Могут быть другие объявления.

4. Вводятся значения: количество объектов; размерность массива; величины допустимого отклонения; количество итераций.

5. В цикле создаются производные объекты, согласно введенному

количеству.

6. Запускается цикл согласно количеству итераций.

6.1. С новой строки выводится: Iteration = «номер итерации»

6.2. С новой строки выводится: The original

6.3. Для каждого объекта выводиться значения элементов массива.

6.4. Вводится значение величины возмущения.

6.5. Для каждого объекта отрабатывает метод возмущения.

6.6. С новой строки выводится: After the outrage

6.7. Посредством объекта анализа и корректировки состояния производного объекта, проверяется и приводиться в нормальный режим работы каждый производный объект.

6.8. С новой строки выводится: The result

6.9. Для каждого объекта выводиться значения элементов массива.

7. После завершения цикла итераций, созданные производные объекты удаляются (уничтожаются).

1.1 Описание входных данных

Первая строка:

`«целое число, количество объектов»`

Вторая строка:

`«целое число, размерность массива»`

Третья строка:

`«целое число, величина допустимого отклонения»`

Четвертая строка:

`«целое число, количество итераций»`

Начиная с пятой строки, значения элементов массивов и имя очередного

объекта, согласно количеству объектов:

```
«целое число» «целое число» . . . «целое число» «строка»
```

Далее, построчно значения величины возмущения, согласно количества итерации.

```
«целое число, значение величины возмущения»
```

Пример ввода

```
2
5
6
4
1 -1 1 -1 1 object_1
-2 2 -2 2 -2 object_2
3
4
2
-7
```

1.2 Описание выходных данных

В процессе каждой итерации производится вывод.

```
Iteration = «номер итерации»
The original
«имя объекта»  «целое число»  «целое число» . . . «целое число»
```

Наименование объекта и значения элементов массива, согласно последовательности создания объектов.

```
After the outrage
«имя объекта»  «целое число»  «целое число» . . . «целое число»
```

Наименование объекта и значения элементов массива, согласно последовательности создания объектов.

```
The result
```

«имя объекта» «целое число» «целое число» . . . «целое число»

Наименование объекта и значения элементов массива, согласно последовательности создания объектов.

Пример вывода

```
Iteration = 1
The original
object_1  1  -1  1  -1  1
object_2  -2  2  -2  2  -2
After the outrage
object_1  4  2  4  2  4
object_2  1  5  1  5  1
The result
object_1  4  2  4  2  4
object_2  1  5  1  5  1
Iteration = 2
The original
object_1  4  2  4  2  4
object_2  1  5  1  5  1
After the outrage
object_1  8  6  8  6  8
object_2  5  9  5  9  5
The result
object_1  0  6  0  6  0
object_2  5  0  5  0  5
Iteration = 3
The original
object_1  0  6  0  6  0
object_2  5  0  5  0  5
After the outrage
object_1  2  8  2  8  2
object_2  7  2  7  2  7
The result
object_1  2  0  2  0  2
object_2  0  2  0  2  0
Iteration = 4
The original
object_1  2  0  2  0  2
object_2  0  2  0  2  0
After the outrage
object_1  -5  -7  -5  -7  -5
object_2  -7  -5  -7  -5  -7
The result
object_1  -5  0  -5  0  -5
object_2  0  -5  0  -5  0
```

2 МЕТОД РЕШЕНИЯ

Для решения задачи используется:

- объект out класса Deriv* предназначен для хранения указателя на один из объектов в коллекции mass_of_objects;
- объект checker класса Checklist* предназначен для хранения указателя на объект, который будет использоваться для проверки;
- библиотека iostream;
- библиотека vector;
- объекты стандартного потока ввода/вывода данных cin/cout;
- оператор присваивания;
- условный оператор <;
- оператор инкремента;
- оператор .;
- оператор ->;
- оператор цикла со счетчиком for;
- условный оператор ветвления if...else;
- арифметический оператор сложения +;
- оператор проверки на равенство ==;
- оператор указателя;
- оператор получения адресса/ссылки.

Класс Original:

- свойства/поля:
    - поле размерность массива:
        - наименование — size;
        - тип — int;
        - модификатор доступа — private;

    - поле максимальное возмущение:
        - наименование — deviation;
        - тип — int;
        - модификатор доступа — private;
    - поле хранит введенные элементы:
        - наименование — mass;
        - тип — *int;
        - модификатор доступа — private;
- функционал:
    - метод Original — параметризированный конструктор;
    - метод disturb — добавляет ко всем элементам поля mass переданное в параметр значение;
    - метод pout — выводит элементы поля mass;
    - метод ~Original — измененный деструктор.

Класс Deriv:

- свойства/поля:
    - поле хранение метода объекта:
        - наименование — name;
        - тип — string;
        - модификатор доступа — private;
- функционал:
    - метод Deriv — параметризированный конструктор;
    - метод out — выводит все элементы поля mass и имя объекта.

Класс Checklist:

- функционал:
    - метод check — проверяет значения всех элементов поля mass переданного объекта.

Таблица 1 – Иерархия наследования классов

№	Имя класса	Классы-наследники	Модификатор доступа при наследовании	Описание	Номер
1	Original			исходный класс	
		Deriv	public		2
2	Deriv			производный класс	
3	Checklist			дружественный класс	

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 Original

Функционал: параметризированный конструктор.

Параметры: целый - sz, размерность массива; целый - dev, максимальное отклонение.

Алгоритм конструктора представлен в таблице 2.

Таблица 2 – Алгоритм конструктора класса Original

№	Предикат	Действия	№ перехода
1		присвоение значению поля size значения параметра sz	2
2		присвоение значению поля deviation значения параметра dev	3
3		выделение памяти под массив mass размера sz	4
4		инициализация переменной i со значением 0	5
5	i < sz?	ввод значения для i элемента массива mass	6
			∅
6		вызов оператора инкремента для i	5

3.2 Алгоритм метода disturb класса Original

Функционал: добавляет ко всем элементам поля mass переданное в параметр значение.

Параметры: целый - dist, добавляет это значение всем элементам поля mass.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода disturb класса Original

№	Предикат	Действия	№ перехода
1		инициализация переменной i со значением 0	2
2	i < size?	добавление значения параметра dist к значению i элемента массива mass	3
			∅
3		вызов оператора инкремента для i	2

3.3 Алгоритм метода pout класса Original

Функционал: выводит элементы поля mass.

Параметры: нет.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода pout класса Original

№	Предикат	Действия	№ перехода
1		инициализация переменной i со значением 0	2
2	i < size?	вывод " ", i элемента массива mass	3
			4
3		вызов оператора инкремента для i	2
4		переход на новую строку	∅

3.4 Алгоритм деструктора класса Original

Функционал: измененный деструктор.

Параметры: нет.

Алгоритм деструктора представлен в таблице 5.

Таблица 5 – Алгоритм деструктора класса Original

№	Предикат	Действия	№ перехода
1		освобождение выделенной под поле mass памяти	∅

3.5 Алгоритм конструктора класса Deriv

Функционал: параметризированный конструктор.

Параметры: целый - sz, размерность массива; целый - dev, максимальное отклонение.

Алгоритм конструктора представлен в таблице 6.

Таблица 6 – Алгоритм конструктора класса Deriv

№	Предикат	Действия	№ перехода
1		ввод значения для поля name	∅

3.6 Алгоритм метода out класса Deriv

Функционал: выводит все элементы поля mass и имя объекта.

Параметры: нет.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода out класса Deriv

№	Предикат	Действия	№ перехода
1		вывод поля name	2
2		вызов метода pout() исходного класса	∅

3.7 Алгоритм метода check класса Checklist

Функционал: проверяет значения всех элементов поля mass переданного объекта.

Параметры: Original&, obj, объект для проверки.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 8.

Таблица 8 – Алгоритм метода check класса Checklist

№	Предикат	Действия	№ перехода
1		инициализация переменной i со значением 0	2
2	i < size?		3
			∅
3	модуль i элемента массива mass объекта obj больше максимального волнения объекта obj?	значение i элемента массива mass прирванивается к нулю	4
			4
4		вызов оператора инкремента для переменной i	2

3.8 Алгоритм функции main

Функционал: главная функция программы.

Параметры: нет.

Возвращаемое значение: целое число, идентификатор работоспособности программы.

Алгоритм функции представлен в таблице 9.

Таблица 9 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		объявление целочисленных переменных amm_of_objects, size, deviation, iter	2
2		объявление строковой переменной name	3
3		ввод значений для переменных amm_of_objects, size, deviation, iter	4
4		создание последовательного контейнера (вектора) mass_of_objects хранящего коллекцию объектов типа Deriv*	5
5		инициализация переменной i со значением 0	6
6	i < amm_of_objects?	добавляем элемент в коллекцию mass_of_objects с помощью метода push_back с параметром, в котором была выделена память под объект класса Deriv, инициализированного с помощью параметризированного конструктора с параметрами size, deviation	7
			8
7		вызов оператора инкремента для переменной i	6
8		объявление целочисленной переменной dev	9
9		инициализация переменной i со значением 1	10
10	i <= iter?	вывод "Iteration = ", значение переменной i, переход на новую строку	11
			29
11		вывод "The original", переход на новую строку	12
12		инициализация переменной j со значением 0	13
13	j < amm_of_objects?	вызов метода out для j элемента коллекции mass_of_objects	14
			15
14		вызов оператора инкремента для переменной j	13

№	Предикат	Действия	№ перехода
15		ввод значения для переменной dev	16
16		инициализация переменной j со значением 0	17
17	j < amm_of_objects?	вызов метода disturb с переданным параметром dev для i элемента коллекцииmass_of_objects	18
			19
18		вызов оператора инкремента для переменной j	17
19		вывод "After the outrage", переход на новую строку	20
20		инициализация переменной j со значением 0	21
21	j < amm_of_objects?	вызов метода out для j элемента коллекции mass_of_objects	22
			23
22		вызов оператора инкремента для переменной j	21
23		вывод "The result", переход на новую строку	24
24		инициализация переменной j со значением 0	25
25	j < amm_of_objects?	объявление объекта checker класса Checklist	26
			10
26		вызов метода check с переданным параметром j элемента коллекции mass_of_objects для объекта checker	27
27		вызов метода out для j элемента коллекции mass_of_objects	28
28		вызов оператора инкремента для переменной j	25
29		инициализация переменной i со значением 0	30
30	i < amm_of_objects?	вызов оператора delete для i элемента коллекции mass_of_objects	31
			∅
31		вызов операторра инкремента для переменной i	30

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-9.

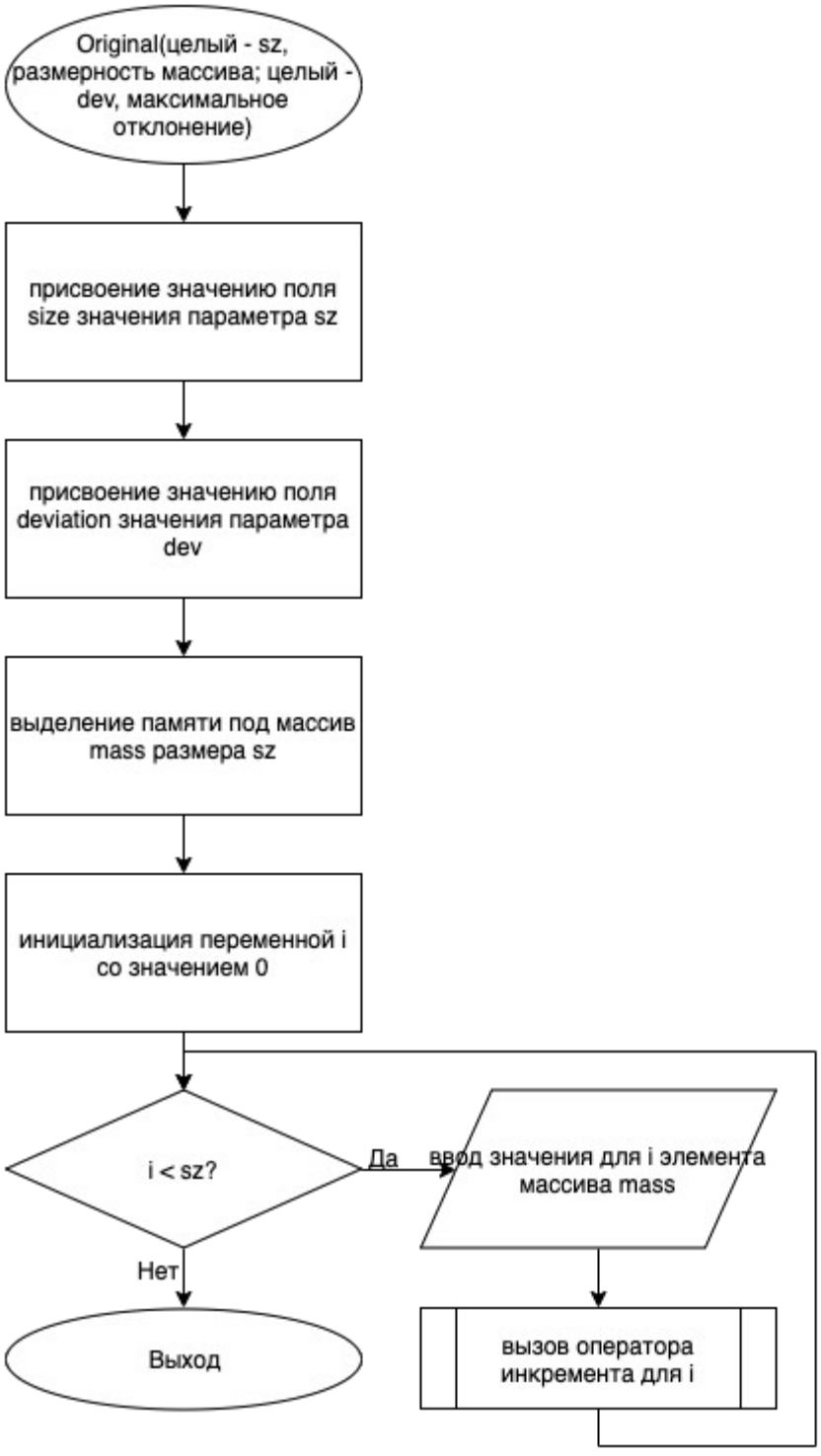

Рисунок 1 – Блок-схема алгоритма

Рисунок 2 – Блок-схема алгоритма

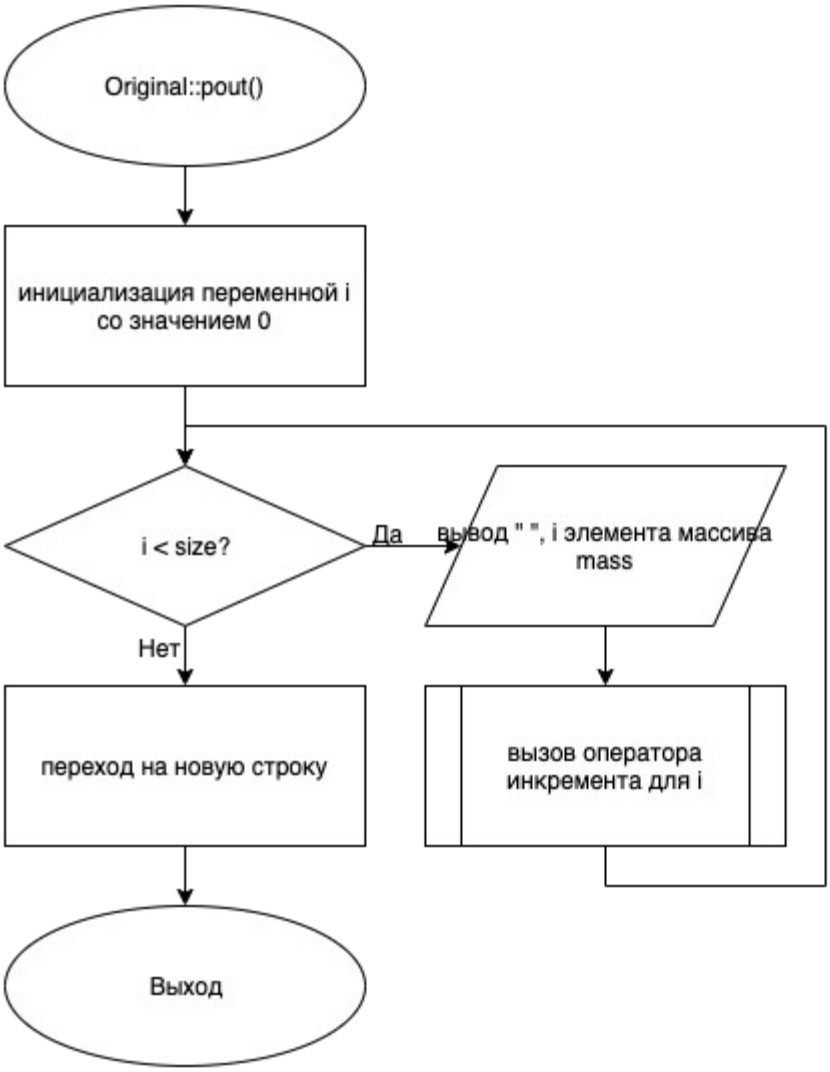

Рисунок 3 – Блок-схема алгоритма

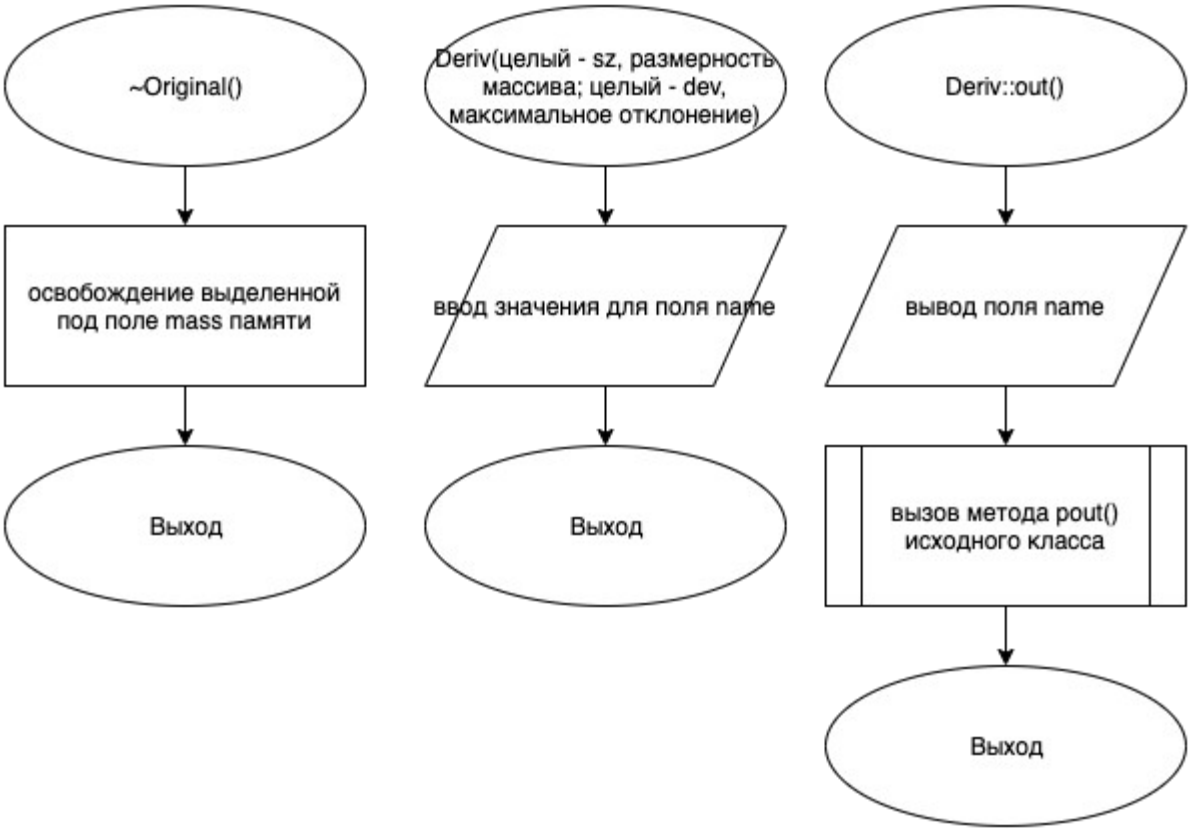

Рисунок 4 – Блок-схема алгоритма

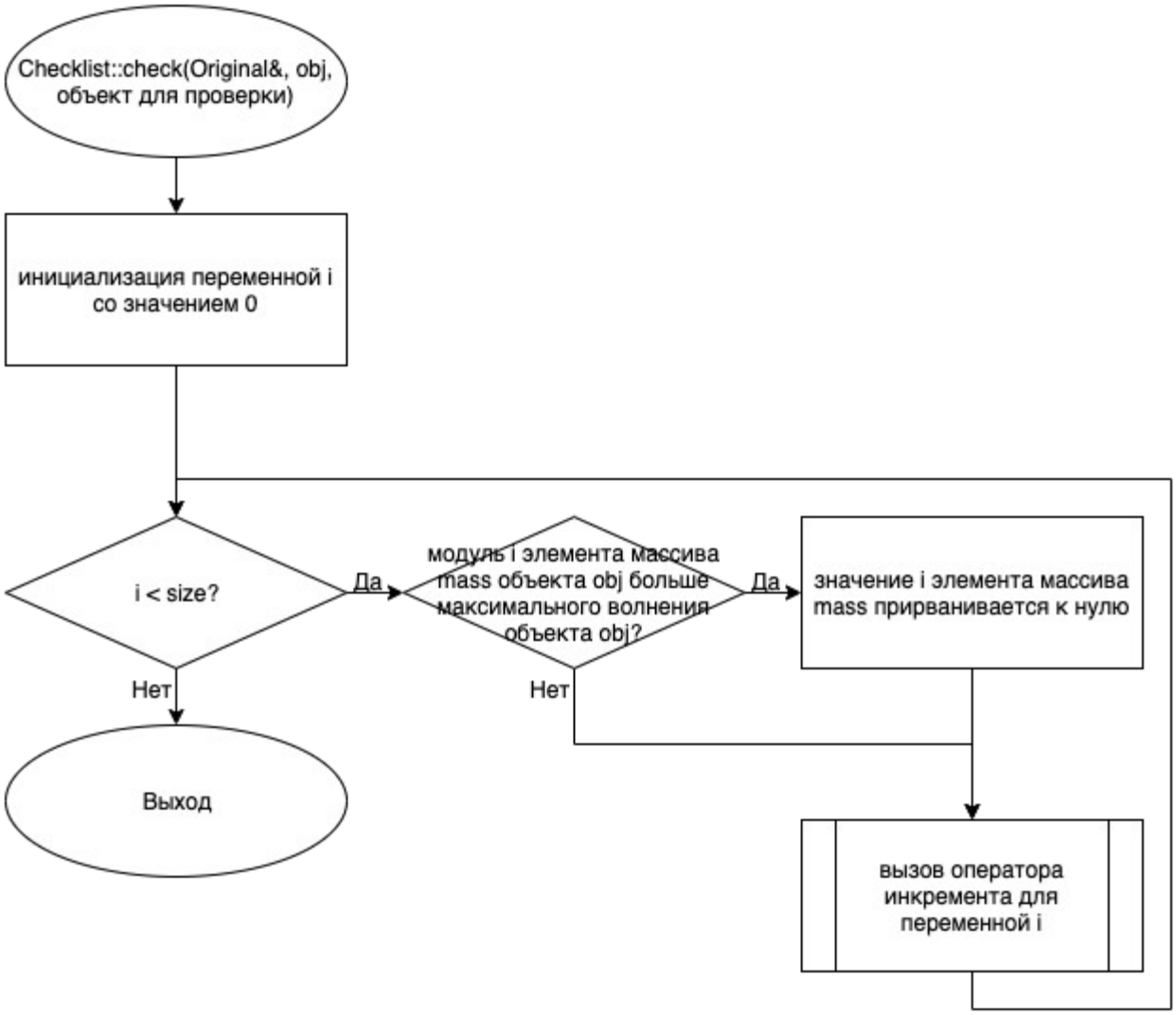

Рисунок 5 – Блок-схема алгоритма

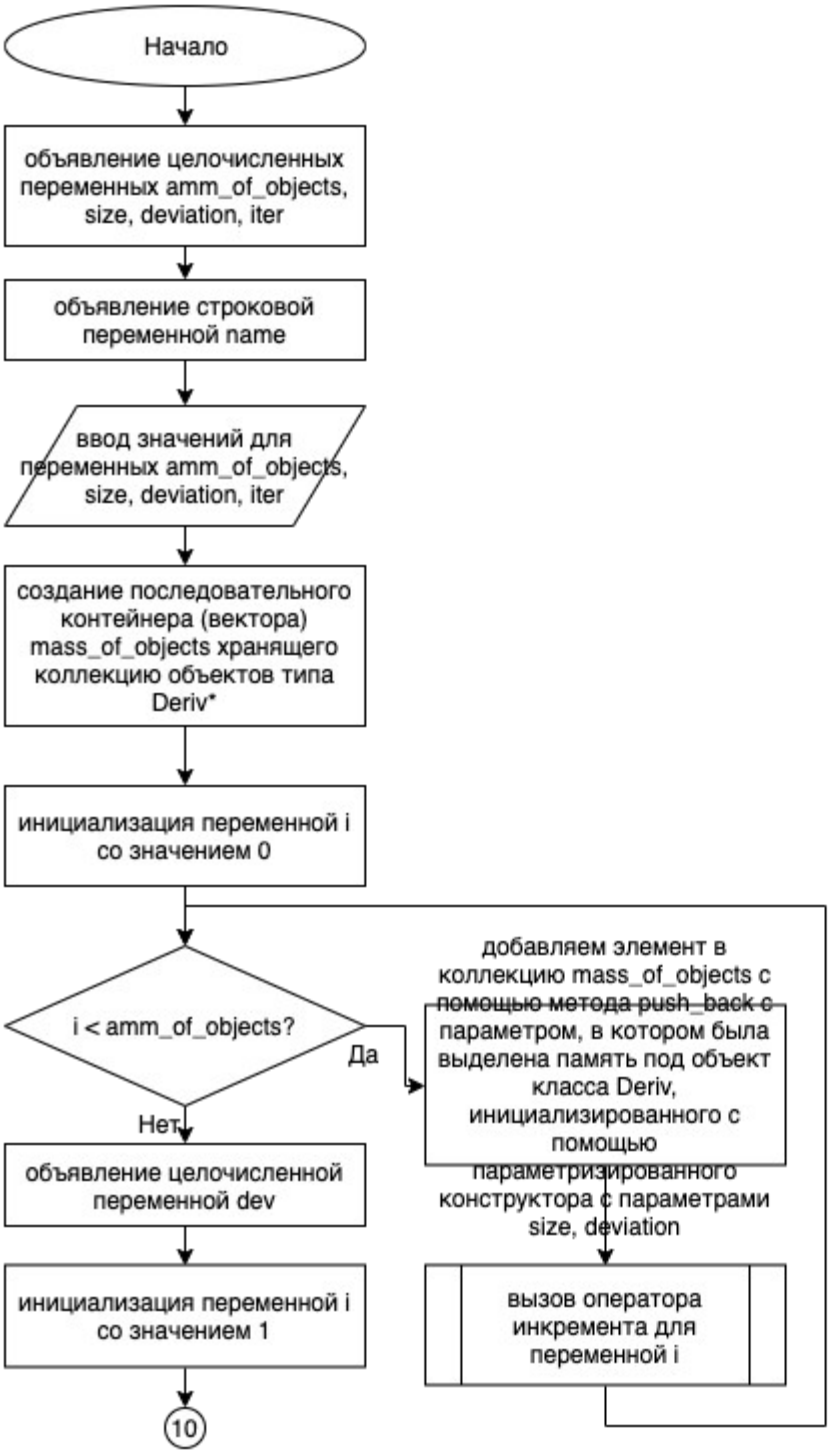

Рисунок 6 – Блок-схема алгоритма

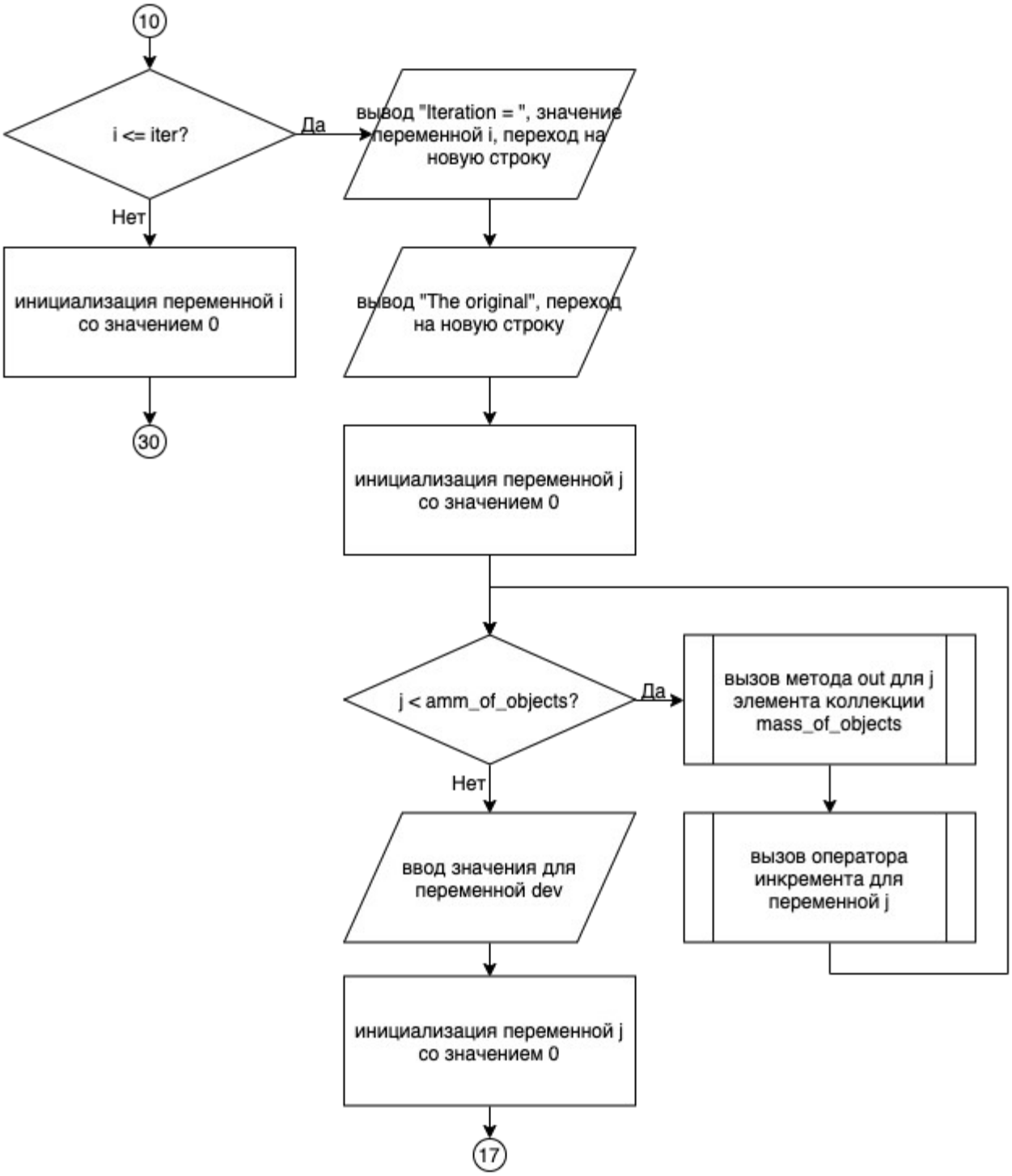

Рисунок 7 – Блок-схема алгоритма

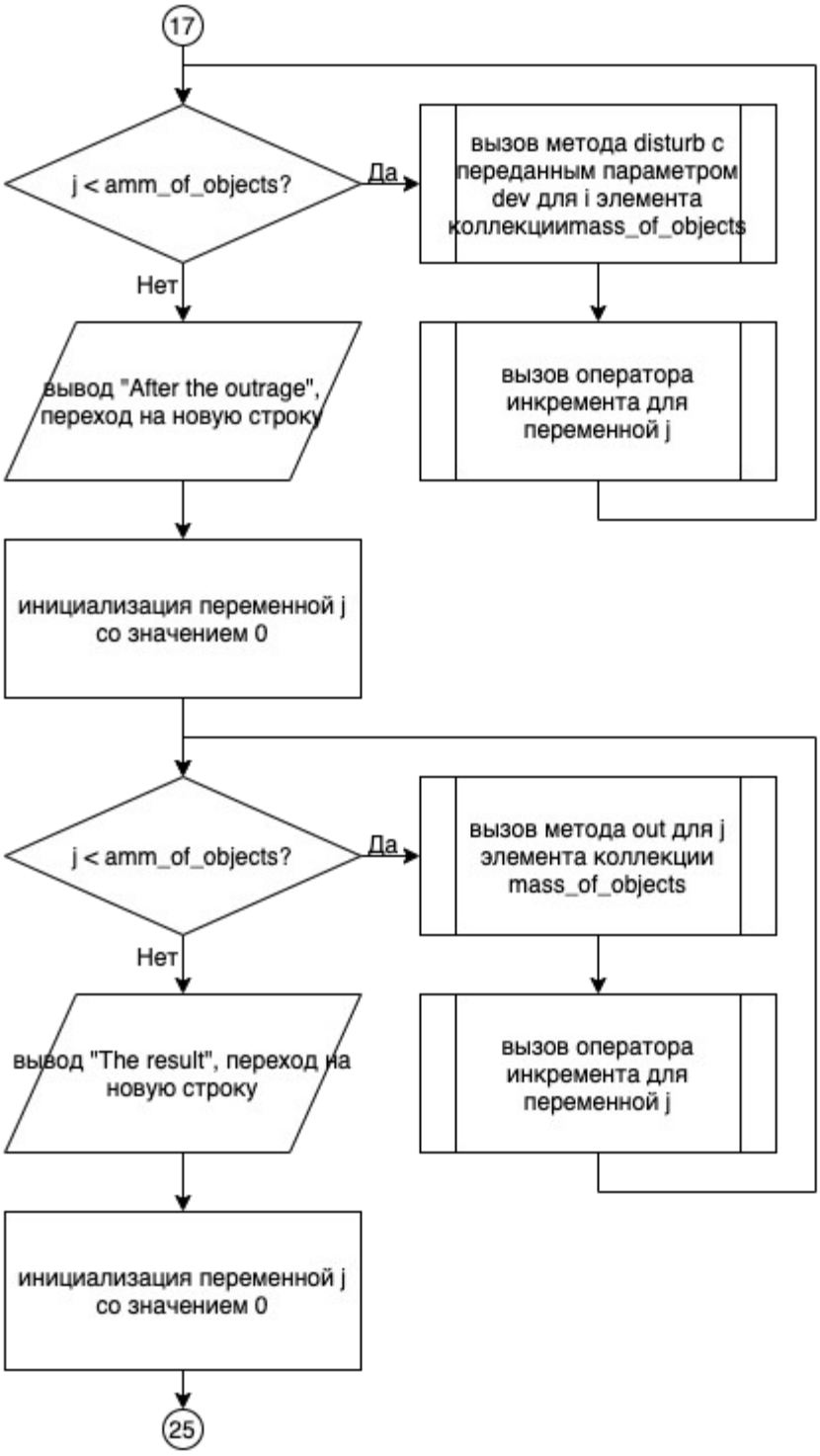

Рисунок 8 – Блок-схема алгоритма

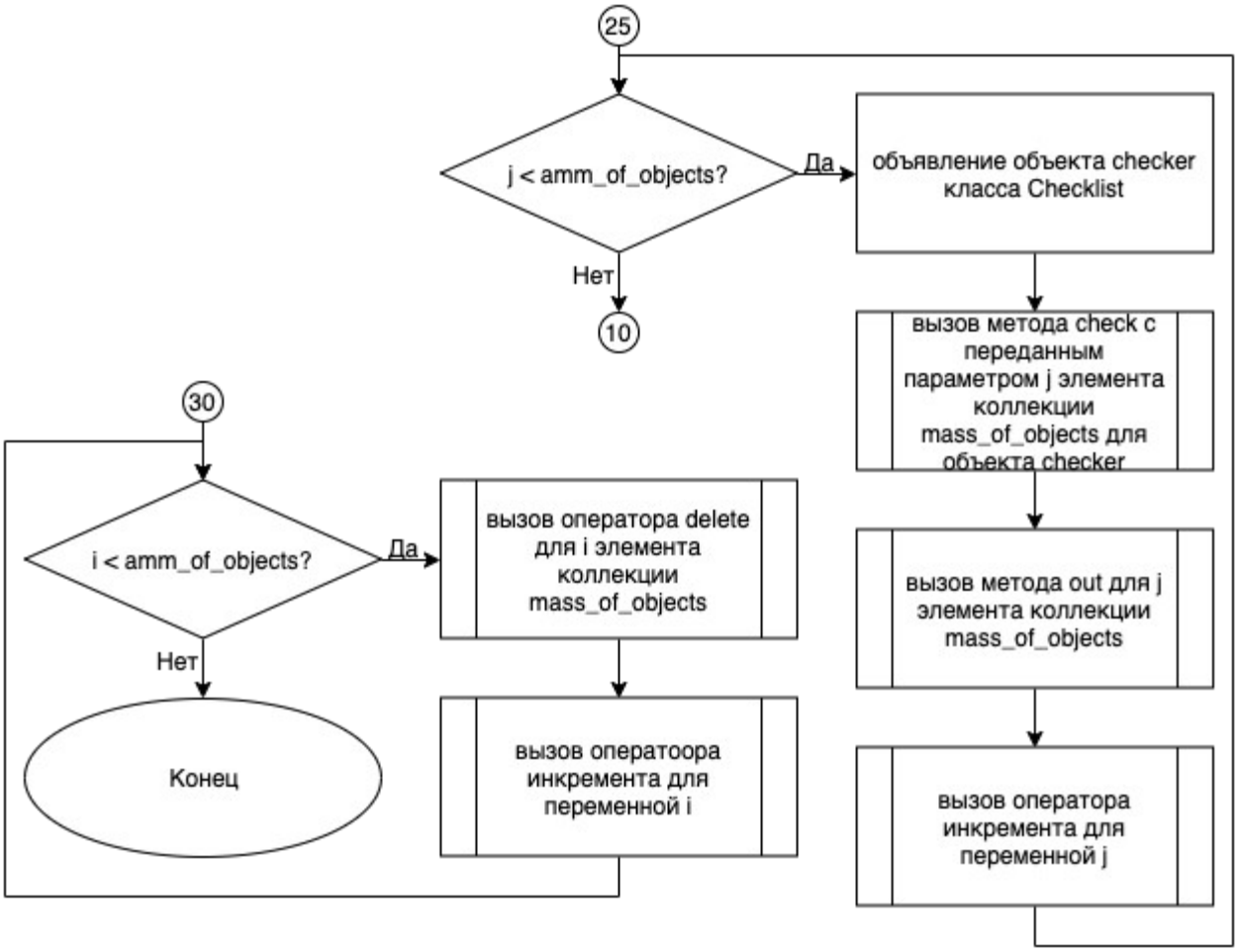

Рисунок 9 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл Checklist.cpp

Листинг 1 – Checklist.cpp

```
#include "Checklist.h"
#include "Original.h"

void Checklist::check(Original& obj){
   for (int i = 0; i < obj.size; i++){
      if (abs(obj.mass[i]) > obj.deviation){
         obj.mass[i] = 0;
      }
   }
}
```

5.2 Файл Checklist.h

Листинг 2 – Checklist.h

```
#ifndef __CHECKLIST__H
#define __CHECKLIST__H
#include "Original.h"

class Checklist{
public:
   void check(Original& obj);
};

#endif
```

5.3 Файл Deriv.cpp

Листинг 3 – Deriv.cpp

```
#include "Deriv.h"

Deriv::Deriv(int sz, int dev) : Original(sz, dev){
   cin >> name;
}

void Deriv::out(){
   cout << name;
   this->pout();
}
```

5.4 Файл Deriv.h

Листинг 4 – Deriv.h

```
#ifndef __DERIV__H
#define __DERIV__H
#include <string>
#include "Original.h"

class Deriv:public Original{
   string name;
public:
   Deriv(int sz, int dev);
   void out();
};

#endif
```

5.5 Файл main.cpp

Листинг 5 – main.cpp

```
#include "Deriv.h"
#include "Checklist.h"

int main()
{
   int amm_of_objects, size, deviation, iter;
```

```
        string name;
        cin >> amm_of_objects >> size >> deviation >> iter;
        vector <Deriv*> mass_of_objects;

        for (int i = 0; i < amm_of_objects; i++){
           mass_of_objects.push_back(new Deriv(size, deviation));
        }

        int dev;
        for (int i = 1; i <= iter; i++){
           cout << "Iteration = " << i << endl;
           cout << "The original" << endl;
           for (int j = 0; j < amm_of_objects; j++){
              mass_of_objects[j]->out();
           }
           cin >> dev;
           for (int j = 0; j < amm_of_objects; j++){
              mass_of_objects[j]->disturb(dev);
           }

           cout << "After the outrage" << endl;
           for (int j = 0; j < amm_of_objects; j++){
              mass_of_objects[j]->out();
           }

           cout << "The result" << endl;
           for (int j = 0; j < amm_of_objects; j++){
              Checklist checker;
              checker.check(*mass_of_objects[j]);
              mass_of_objects[j]->out();
           }
        }
        for (int i = 0; i < amm_of_objects; i++){
           delete mass_of_objects[i];
        }

        return(0);
     }
```

5.6 Файл Original.cpp

Листинг 6 – Original.cpp

```
     #include "Original.h"

     Original::Original(int sz, int dev = 0){
        size = sz;
        deviation = dev;
        mass = new int[sz];
        for (int i = 0; i < sz; i++){
```

```
      cin >> mass[i];
      }
   }
   void Original::pout(){
      for (int i = 0; i < size; i++){
         cout << "  " << mass[i];
      }
      cout << endl;
   }
   void Original::disturb(int dist){
      for (int i = 0; i < size; i++){
         mass[i] += dist;
      }
   }
   Original::~Original(){
      delete[] mass;
   }
```

5.7 Файл Original.h

Листинг 7 – Original.h

```
   #ifndef __ORIGINAL__H
   #define __ORIGINAL__H
   #include <iostream>
   #include <vector>
   using namespace std;

   class Original{
      friend class Checklist;
      int size;
      int deviation;
      int* mass;
   public:
      Original(int sz, int dev);
      void disturb(int dist);
      void pout();
      virtual ~Original();
   };

   #endif
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 10.

Таблица 10 – Результат тестирования программы

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
2 5 6 4 1 -1 1 -1 1 object_1 -2 2 -2 2 -2 object_2 3 4 2 -7	Iteration = 1 The original object_1 1 -1 1 -1 1 object_2 -2 2 -2 2 -2 After the outrage object_1 4 2 4 2 4 object_2 1 5 1 5 1 The result object_1 4 2 4 2 4 object_2 1 5 1 5 1 Iteration = 2 The original object_1 4 2 4 2 4 object_2 1 5 1 5 1 After the outrage object_1 8 6 8 6 8 object_2 5 9 5 9 5 The result object_1 0 6 0 6 0 object_2 5 0 5 0 5 Iteration = 3 The original object_1 0 6 0 6 0 object_2 5 0 5 0 5 After the outrage object_1 2 8 2 8 2 object_2 7 2 7 2 7 The result object_1 2 0 2 0 2	Iteration = 1 The original object_1 1 -1 1 -1 1 object_2 -2 2 -2 2 -2 After the outrage object_1 4 2 4 2 4 object_2 1 5 1 5 1 The result object_1 4 2 4 2 4 object_2 1 5 1 5 1 Iteration = 2 The original object_1 4 2 4 2 4 object_2 1 5 1 5 1 After the outrage object_1 8 6 8 6 8 object_2 5 9 5 9 5 The result object_1 0 6 0 6 0 object_2 5 0 5 0 5 Iteration = 3 The original object_1 0 6 0 6 0 object_2 5 0 5 0 5 After the outrage object_1 2 8 2 8 2 object_2 7 2 7 2 7 The result object_1 2 0 2 0

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
	object_2 0 2 0 2 0 Iteration = 4 The original object_1 2 0 2 0 2 object_2 0 2 0 2 0 After the outrage object_1 -5 -7 -5 -7 -5 object_2 -7 -5 -7 -5 -7 The result object_1 -5 0 -5 0 -5 object_2 0 -5 0 -5 0	2 object_2 0 2 0 2 0 Iteration = 4 The original object_1 2 0 2 0 2 object_2 0 2 0 2 0 After the outrage object_1 -5 -7 -5 -7 -5 object_2 -7 -5 -7 -5 -7 The result object_1 -5 0 -5 0 -5 object_2 0 -5 0 -5 0

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).