Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

Дан объект следующей конструкции:

В закрытом доступе имеется массив целого типа и поле его длины. Количество элементов массива четное и больше двух. Объект имеет функциональность:

- конструктор по умолчанию, вначале работы выдает сообщение;
- параметризированный конструктор, передается целочисленный параметр. Параметр должен иметь значение больше 2 и быть четным. Вначале работы выдает сообщение;
- конструктор копии, обеспечивает создание копии объекта в новой области памяти. Вначале работы выдает сообщение;
- метод деструктор, который в начале работы выдает сообщение;
- метод который создает целочисленный массив в закрытой области, согласно ранее заданной размерности.
- метод ввода значений элементов созданного массива;
- метод 1, который суммирует значения очередной пары элементов и сумму присваивает первому элементу пары. Например, пусть массив состоит из элементов {1,2,3,4}. В результате суммирования пар получим массив {3,2,7,4};
- метод 2, который умножает значения очередной пары элементов и результат присваивает первому элементу пары. Например, пусть массив состоит из элементов {1,2,3,4}. В результате умножения пар получим массив {2,2,12,4};
- метод, который суммирует значения элементов массива и возвращает это значение;
- метод последовательного вывода содержимого элементов массива,

которые разделены двумя пробелами;

- метод, который возвращает значение указателя на массив из закрытой области;
- метод, который присваивает значение указателя массива из закрытой области.

Назовём класс описания данного объекта cl_obj (для примера, у вас он может называться иначе).

Разработать функцию func, которая имеет один целочисленный параметр, содержащий размерность массива. В функции должен быть реализован алгоритм:

1. Инициализация указателя на объект класса cl_obj адресом объекта, созданного с использованием параметризированного конструктора.
2. С использованием указателя на объект класса cl_obj вызов метода создания массива.
3. С использованием указателя на объект класса cl_obj вызов метода ввода значений элементов массива.
4. С использованием указателя на объект класса cl_obj вызов метода 2.
5. Возврат указателя на объект класса cl_obj.

В основной функции реализовать алгоритм:

1. Ввод размерности массива.
2. Если размерность массива некорректная, вывод сообщения и завершить работу алгоритма.
3. Вывод значения размерности массива.
4. Объявить первый указатель на объект класса cl_obj.
5. Присвоение первому указателю результата работы функции func с аргументом, содержащим значение размерности массива.
6. С использованием первого указателя вызов метода 1.
7. Инициализация второго указателя на объект класса cl_obj адресом

объекта, созданного с использованием конструктора копии с аргументом первого объекта.

8. С использованием второго указателя вызов метода 2.
9. Вывод содержимого массива первого объекта.
10. Вывод суммы элементов массива первого объекта.
11. Вывод содержимого массива второго объекта.
12. Вывод суммы элементов массива второго объекта.
13. Второму объекту присвоить первый объект.
14. С использованием первого указателя вызов метода 1.
15. Вывод содержимого массива второго объекта.
16. Вывод суммы элементов массива второго объекта.
17. Удалит первый объект.
18. Удалить второй объект.

Добавить в этот алгоритм пункты, которые обеспечат корректное завершение работы программы.

1.1 Описание входных данных

Первая строка:

«Целое число»

Вторая строка:

«Целое число» «Целое число» . . .

Пример:

```
4
3 5 1 2
```

1.2 Описание выходных данных

Если введенная размерность массива допустима, то в первой строке выводится это значение:

```
«Целое число»
```

Если введенная размерность массива не больше двух или нечетная, то в первой строке выводится некорректное значение и вопросительный знак:

```
«Целое число»?
```

Конструктор по умолчанию в начале работы с новой строки выдает сообщение:

```
Default constructor
```

Параметризированный конструктор в начале работы с новой строки выдает сообщение:

```
Constructor set
```

Конструктор копии в начале работы с новой строки выдает сообщение:

```
Copy constructor
```

Деструктор в начале работы с новой строки выдает сообщение:

```
Destructor
```

Метод последовательного вывода содержимого элементов массива, с новой строки выдает:

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

Пример вывода:

```
4
Constructor set
Copy constructor
20  5  4  2
31
100  5  8  2
```

```
115
100  5  8  2
115
Destructor
Destructor
```

2 МЕТОД РЕШЕНИЯ

Для решения задачи используется:

- функция fun для работы с указателем на объект;
- Объект стандартного потока ввода с клавиатуры cin;
- Объект стандартного потока вывода на экран cout;
- Условный оператор if..else;
- Оператор цикла for.

Класс Test:

- свойства/поля:
  - поле Размер массива:
    - наименование — n;
    - тип — int;
    - модификатор доступа — private;
  - поле Указатель на массив:
    - наименование — mass;
    - тип — int*;
    - модификатор доступа — private;
- функционал:
  - метод output — метод последовательного вывода содержимого элементов массива, которые разделены двумя пробелами;
  - метод get_mass — метод, который возвращает значение указателя на массив из закрытой области;
  - метод put_mass — метод, который присваивает значение указателя массива из закрытой области.

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 метода output класса Test

Функционал: метод последовательного вывода содержимого элементов массива, которые разделены двумя пробелами.

Параметры: нет.

Возвращаемое значение: Ничего.

Алгоритм метода представлен в таблице 1.

Таблица 1 – Алгоритм метода output класса Test

№	Предикат	Действия	№ перехода
1		Объявление целочисленной переменной i и инициализация 0	2
2	i<n-1	Вывод на экран значения ячейки с адресом mass[i] и двух пробелов	3
		Вывод на экран значения ячейки mass[n-1] и переноса строки	∅
3		Увеличение i на 1	2

3.2 Алгоритм метода get_mass класса Test

Функционал: метод, который возвращает значение указателя на массив из закрытой области.

Параметры: нет.

Возвращаемое значение: Указатель на массив из закрытой облати.

Алгоритм метода представлен в таблице 2.

Таблица 2 – Алгоритм метода get_mass класса Test

№	Предикат	Действия	№ перехода
1		Вернуть значение указателя mass	∅

3.3 Алгоритм метода put_mass класса Test

Функционал: метод, который присваивает значение указателя массива из закрытой области.

Параметры: Указатель на целочисленный массив.

Возвращаемое значение: Ничего.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода put_mass класса Test

№	Предикат	Действия	№ перехода
1		Указателю mass присваиваем значение параметра arr	∅

3.4 Алгоритм функции fun

Функционал: работы с указателем на объект.

Параметры: Целочисленный параметр x обозначающий размер массива.

Возвращаемое значение: Указатель на объект.

Алгоритм функции представлен в таблице 4.

Таблица 4 – Алгоритм функции fun

№	Предикат	Действия	№ перехода
1		Инициализация указателя local на объект класса Test, созданного с параметром x	2

№	Предикат	Действия	№ перехода
2		Вызов метода new_mass объекта по адресу local	3
3		Вызов метода input объекта по адресу local	4
4		Вызов метода metod2 объекта по адресу local	5
5		Возврат local	∅

3.5 Алгоритм функции main

Функционал: Выполнение действий, описанных в задаче.

Параметры: нет.

Возвращаемое значение: Целочисленное значение.

Алгоритм функции представлен в таблице 5.

Таблица 5 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		Объявление целочисленной переменной x	2
2		Ввод с клавиатуры значения x	3
3	x>2 и x четно	Вывод на экран "(x)"	4
		Вывод на экран "(x)?"	∅
4		Создание указателя obj1 на объект класса Test	5
5		Вызов функции fun с параметром x и инициализация obj1 результатом	6
6		Инициализация указателя obj2 адресом объекта, созданного с конструктором копии с параметром obj1	7
7		Вызов метода motod2 объекта по адресу obj2	8
8		Вызов метода output объекта по адресу obj1	9
9		Вызов метода sum объекта по адресу obj1 и вывод на экран	10
10		Вызов метода output объекта по адресу obj2	11

№	Предикат	Действия	№ перехода
11		Вызов метода sum объекта по адресу obj2 и вывод на экран	12
12		Инициализация указателя arr значением, возвращаемым методом get_mass	13
13		obj2 присваивается obj1 по значению	14
14		Вызов метода put_mass объекта по адресу obj2 с параметром arr	15
15		Вызов метода metod1 объекта по адресу obj1	16
16		Вызов метода output объекта по адресу obj2	17
17		Вызов метода sum объекта по адресу obj2 и вывод на экран	18
18		Удаление obj1	19
19		Удаление obj2	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-3.

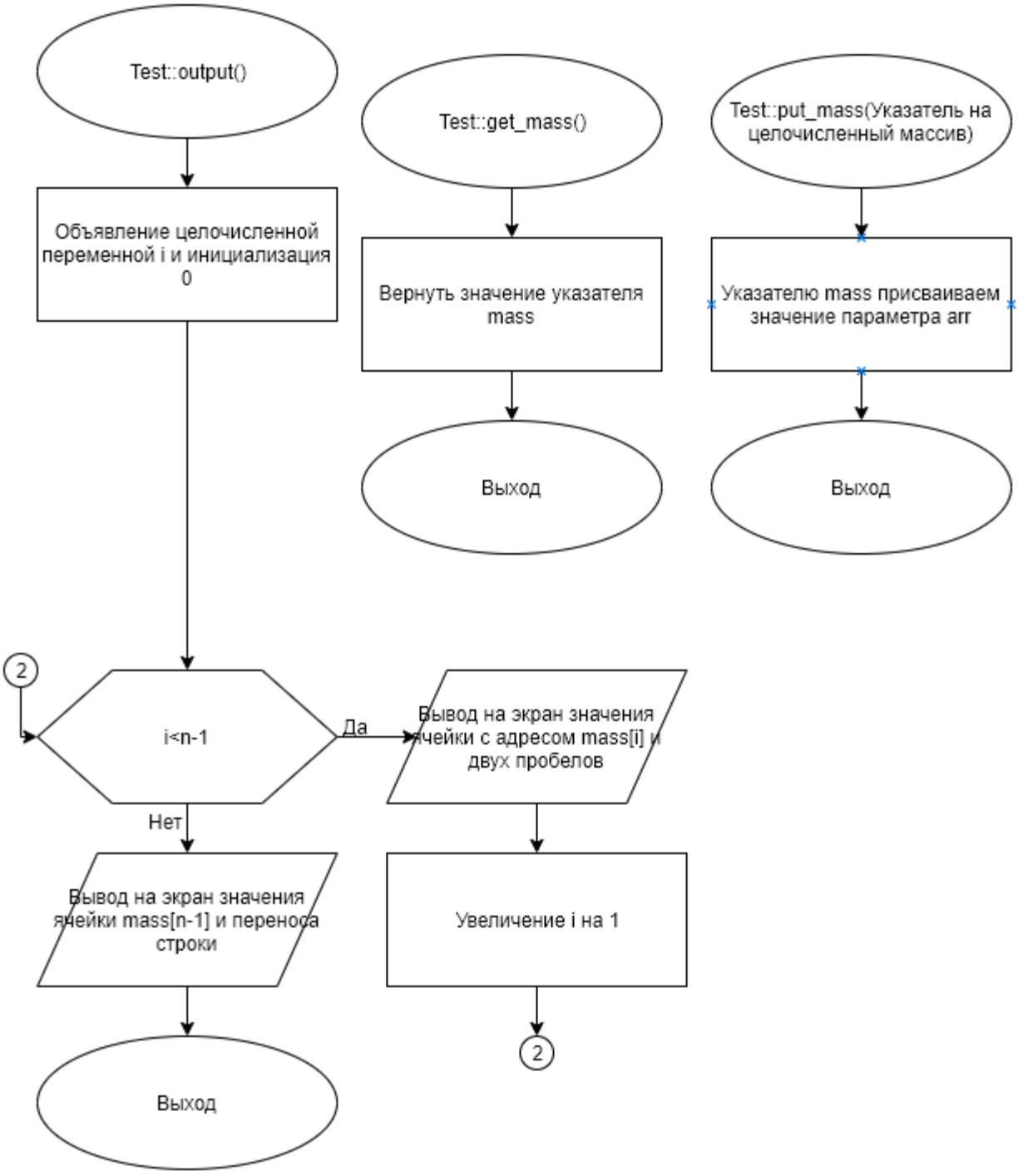

Рисунок 1 – Блок-схема алгоритма

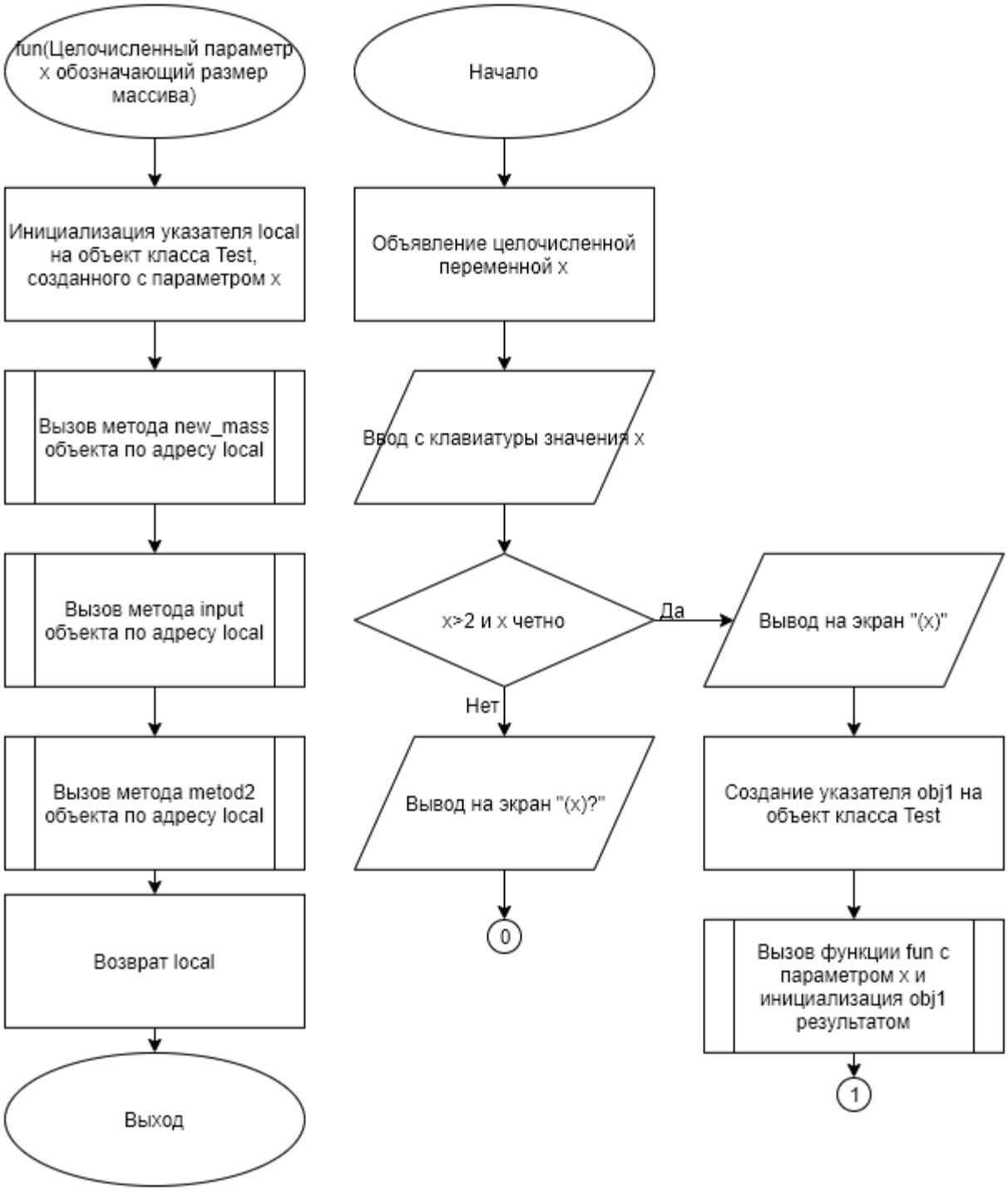

Рисунок 2 – Блок-схема алгоритма

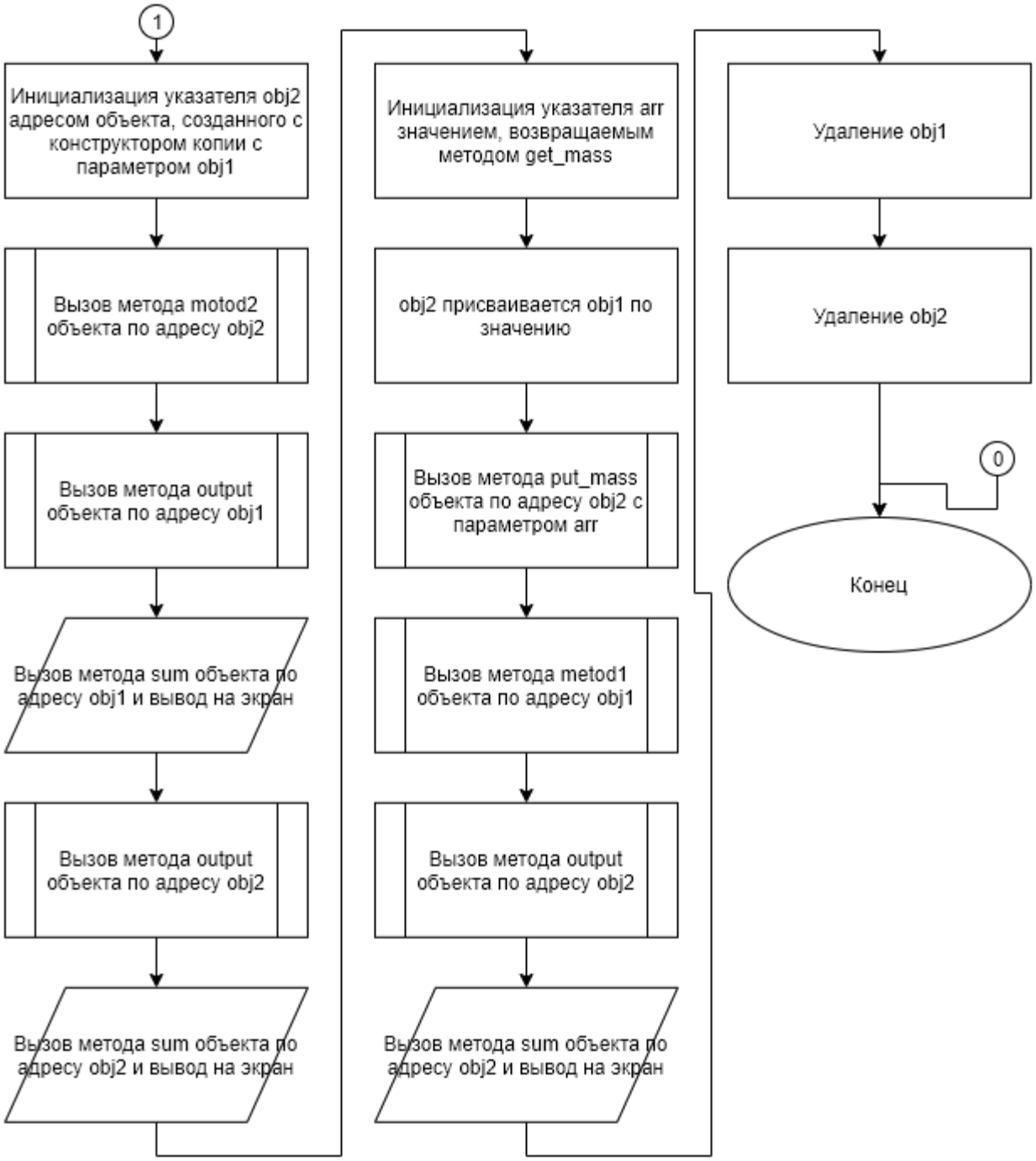

Рисунок 3 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл main.cpp

Листинг 1 – main.cpp

```
#include <stdlib.h>
#include <stdio.h>
#include <iostream>
#include "Test.h"

using namespace std;

Test* func(int x)
{
   Test* local = new Test(x);
   local->new_mass();
   local->input();
   local->metod2();
   return local;
}

int main()
{
   int x;
   cin >> x;
   if (x > 2 && x % 2 == 0) {
      cout << x << endl;
      Test* obj1;
      obj1 = func(x);
      obj1->metod1();
      Test* obj2 = new Test(*obj1);
      obj2->metod2();
      obj1->output();
      cout << obj1->sum() << endl;
      obj2->output();
      cout << obj2->sum() << endl;
      int* arr = obj2->get_mass();
      *obj2 = *obj1;
      obj2->put_mass(arr);
      obj1->metod1();
      obj2->output();
      cout << obj2->sum() << endl;
      delete obj1;
      delete obj2;
```

```
   }
   else {
      cout << x << "?";
      exit(0);
   }
}
```

5.2 Файл Test.cpp

Листинг 2 – Test.cpp

```
#include "Test.h"

using namespace std;

Test::Test()
{
   cout << "Default constructor" << endl;
}

Test::Test(int x)
{
   cout << "Constructor set" << endl;
   n = x;
}

Test::Test(const Test & ob)
{
   cout << "Copy constructor" << endl;
   n = ob.n;
   mass = new int[n];
   for(int i = 0; i < n; i++) {
      mass[i] = ob.mass[i];
   }
}

void Test::new_mass()
{
   mass = new int[n];
}

void Test::input()
{
   for(int i = 0; i < n; i++) {
      cin >> mass[i];
   }
}

void Test::output()
{
```

```
   for(int i = 0; i < n - 1; i++) {
      cout << mass[i] << "  ";
   }
   cout << mass[n - 1] << endl;
}

int Test::sum()
{
   int summ = 0;
   for (int i = 0; i < n; i++) {
      summ += mass[i];
   }
   return(summ);
}

void Test::metod1()
{
   for (int i = 0; i < n; i += 2) {
      mass[i] += mass[i + 1];
   }
}

void Test::metod2()
{
   for (int i = 0; i < n; i += 2) {
      mass[i] *= mass[i + 1];
   }
}

int* Test::get_mass()
{
   return mass;
}

void Test::put_mass(int* arr)
{
   mass = arr;
}

Test::~Test()
{
   cout << "Destructor" << endl;
}
```

5.3 Файл Test.h

Листинг 3 – Test.h

```
#ifndef __TEST__H
#define __TEST__H
```

```
#include <iostream>

using namespace std;

class Test
{
private:
int n;
int* mass;

public:
Test();
Test(int x);
Test(const Test & ob);
void new_mass();
void input();
void output();
int sum();
void metod1();
void metod2();
int* get_mass();
void put_mass(int* arr);
~Test();

};


#endif
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 6.

Таблица 6 – Результат тестирования программы

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
4 3 5 1 2	4 Constructor set Copy constructor 20 5 4 2 31 100 5 8 2 115 100 5 8 2 115 Destructor Destructor	4 Constructor set Copy constructor 20 5 4 2 31 100 5 8 2 115 100 5 8 2 115 Destructor Destructor
6 1 2 3 4 5 6	6 Constructor set Copy constructor 4 2 16 4 36 6 68 8 2 64 4 216 6 300 8 2 64 4 216 6 300 Destructor Destructor	6 Constructor set Copy constructor 4 2 16 4 36 6 68 8 2 64 4 216 6 300 8 2 64 4 216 6 300 Destructor Destructor
5 1 2 3 4 5	5?	5?
2 1 2	2?	2?

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).