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, в начале работы выдает сообщение;
- Параметризированный конструктор, передается целочисленный параметр. По значению параметра определяется размерность целочисленного массива из закрытой области. Массив создается. В начале работы выдает сообщение;
- Метод деструктор, который выдает сообщение что он отработал;
- Метод ввода данных для созданного массива;
- Метод 1, который суммирует значения очередной пары элементов и сумму присваивает первому элементу пары. Далее суммирует элементы полученного массива и возвращает это значение. 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. Далее суммирует элементы полученного массива и возвращает это значение. 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.

Разработать функцию, которая в качестве параметра получает объект по значению. Функция вызывается метод 2, далее выводит сумму элементов массива с новой строки.

В основной функции реализовать алгоритм:

1. Ввод размерности массива. Размер должен иметь значение больше 2 и быть четным.
2. Если размерность массива некорректная, вывод сообщения и завершить работу алгоритма.
3. Вывод значения размерности массива.
4. Создание объекта с аргументом размерности массива.
5. Вызов метода для ввода значений элементов массива.
6. Вызов функции передача в качестве аргумента объекта.
7. Вызов метода 1 от имени объекта.
8. Вывод суммы элементов массива объекта с новой строки.

Разработать конструктор копии объекта для корректного выполнения вычислений. В начале работы конструктор копии выдает сообщение с новой строки.

1.1 Описание входных данных

Первая строка:

«Целое число»

Вторая строка:

«Целое число» «Целое число» . . .

Пример:

```
8
1 2 3 4 5 6 7 8
```

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 копирования в начале работы с новой строки выдает сообщение:

```
Copy constructor
```

Деструктор в начале работы с новой строки выдает сообщение:

```
Destructor
```

Пример вывода:

```
8
Constructor set
Copy constructor
120
Destructor
56
Destructor
```

2 МЕТОД РЕШЕНИЯ

Для решения задачи используется:

- объект object класса MyClass предназначен для демонстрации передачи объекта функциями в качестве параметров;
- объект cout класса iostream предназначен для потока вывода на экран;
- объект cin класса iostream предназначен для потока ввода;
- функция Call для вывода результата работы метода MySpace() объекта object;
- if...else - условный оператор;
- for - оператор цикла со счетчиком.

Класс MyClass:

- свойства/поля:
    - поле размера массива masive:
        - наименование — m;
        - тип — int;
        - модификатор доступа — protected;
    - поле целочисленного массива masiive:
        - наименование — masive;
        - тип — int;
        - модификатор доступа — protected;
- функционал:
    - метод MyClass — конструктор по умолчанию, в начале работы выдаёт сообщение об отработке;
    - метод MyClass(int m) — параметризованный конструктор, передаётся целочисленный параметр. По значению параметра , определяется размерность целочисленного массива из закрытой

области;

- метод MyClass(const Class& object) — конструктор копии для исключения ошибки при передаче объекта в функцию по значению;
- метод ~MyClass — деструктор , выводит сообщение об отработке;
- метод Ellement — метод для ввода значений элементов массива;
- метод Sumator — метод , который суммирует значения очередной пары элементов и присваевает сумму первому элементу пары;
- метод MySpace — метод , который умножает значения очередной пары элементов и результат присваевает первому элементу пары , далее суммирует элементы полученного массива и возвращает это значение;
- метод Sum — суммирует значения элементов массива.

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 MyClass

Функционал: конструктор по умолчанию, в начале работы выдаёт сообщение об отработке.

Параметры: нет.

Алгоритм конструктора представлен в таблице 1.

Таблица 1 – Алгоритм конструктора класса MyClass

№	Предикат	Действия	№ перехода
1		вывод "Default constructor" и перенос на новую строку	∅

3.2 Алгоритм конструктора класса MyClass

Функционал: параметризованный конструктор, передаётся целочисленный параметр. По значению параметра , определяется размерность целочисленного массива из закрытой области.

Параметры: int m.

Алгоритм конструктора представлен в таблице 2.

Таблица 2 – Алгоритм конструктора класса MyClass

№	Предикат	Действия	№ перехода
1		вывод "Constructor set" и перенос на новую строку	2
2		создание целочисленного массива masive размерности m	3

№	Предикат	Действия	№ перехода
3		Присвоение значению переменной класса m класса MyClass значения параметра m , поданного в конструктор	∅

3.3 Алгоритм конструктора класса MyClass

Функционал: конструктор копии для исключения ошибки при передаче объекта в функцию по значению.

Параметры: Class& object.

Алгоритм конструктора представлен в таблице 3.

Таблица 3 – Алгоритм конструктора класса MyClass

№	Предикат	Действия	№ перехода
1		вывод "Copy Constructor" и перенос на новую строку	2
2		копирование значение переменной m	3
3		Объявление целочисленного массива masive размерности m	4
4		инициализация целочисленного переменной i со значением 0	5
5	i<m	копирование i элемента массива masive	6
			∅
6		инкремент i	5

3.4 Алгоритм деструктора класса MyClass

Функционал: деструктор , выводит сообщение об отработке.

Параметры: нет.

Алгоритм деструктора представлен в таблице 4.

Таблица 4 – Алгоритм деструктора класса MyClass

№	Предикат	Действия	№ перехода
1		удаление массива по адресу указателя masive	∅

3.5 Алгоритм метода Ellement класса MyClass

Функционал: метод для ввода значений элементов массива.

Параметры: нет.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода Ellement класса MyClass

№	Предикат	Действия	№ перехода
1		объявление переменно x тип int	2
2		инициализация переменной i типа int со значением 0	3
3	i<m	ввод значения переменной x	4
			∅
4		присваивание значению i элемента массива masive значения x	5
5		инкремент i	∅

3.6 Алгоритм метода Sumator класса MyClass

Функционал: метод , который суммирует значения очередной пары элементов и присваевает сумму первому элементу пары.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 6.

Таблица 6 – Алгоритм метода Sumator класса MyClass

№	Предикат	Действия	№ перехода
1		инициализация переменной i типа int со значением 0	2
2	i<m	присваивание значению i элемента массива masive значения суммы i элемента и следующего	3
		возврат результата работы sumator() текущего объекта	∅
3		i+=2	2

3.7 Алгоритм метода MySpace класса MyClass

Функционал: метод , который умножает значения очередной пары элементов и результат присваевает первому элементу пары , далее суммирует элементы полученного массива и возвращает это значение.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода MySpace класса MyClass

№	Предикат	Действия	№ перехода
1		инициализация переменной i типа int со значением 0	2
2	i<m	присваивание значению i элемента массива masive значения произведения i элемента и следующего	3
			∅
3		i+=2	2

3.8 Алгоритм метода Sum класса MyClass

Функционал: суммирует значения элементов массива.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 8.

Таблица 8 – Алгоритм метода Sum класса MyClass

№	Предикат	Действия	№ перехода
1		инициализация переменной s типа int со значением 0	2
2		инициализация переменной i типа int со значением 0	3
3	i<m	значение s увеличивается на значение i элемента массива masive	4
		возврат значения переменной s	∅
4		инкремент i	3

3.9 Алгоритм функции Call

Функционал: вывод результата работы метода MySpace объекта Object.

Параметры: Class object.

Возвращаемое значение: нет.

Алгоритм функции представлен в таблице 9.

Таблица 9 – Алгоритм функции Call

№	Предикат	Действия	№ перехода
1		вывод значения , возвращаемого методом MySpace() для объекта object	∅

3.10 Алгоритм функции main

Функционал: запуск программы.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм функции представлен в таблице 10.

Таблица 10 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		объявление переменной back типа int	2
2		ввод значения переменной back	3
3	back<=2||back%2!=0	вывод значения переменной back и "?"	12
			4
4		вывод значения переменной back	5
5		вывод переноса на новую строку	6
6		создание объекта object класса MyClass с параметром back	7
7		вызов метода Ellement объекта object	8
8		вызов функции Call(object)	9
9		вывод переноса на новую строку	10
10		вызов метода sumator () объекта object	11
11		вывод значения возвратимого методом sum() объекта object	12
12		return 0	∅

3.11 Алгоритм метода MyClass(int m) класса MyClass

Функционал: параметризованный конструктор, передаётся целочисленный параметр. По значению параметра , определяется размерность целочисленного массива из закрытой области.

Параметры: неи.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 11.

Таблица 11 – Алгоритм метода MyClass(int m) класса MyClass

№	Предикат	Действия	№ перехода
1			∅

3.12 Алгоритм метода MyClass(const Class& object) класса MyClass

Функционал: конструктор копии для исключения ошибки при передаче объекта в функцию по значению.

Параметры: нет.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 12.

Таблица 12 – Алгоритм метода MyClass(const Class& object) класса MyClass

№	Предикат	Действия	№ перехода
1			∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-8.

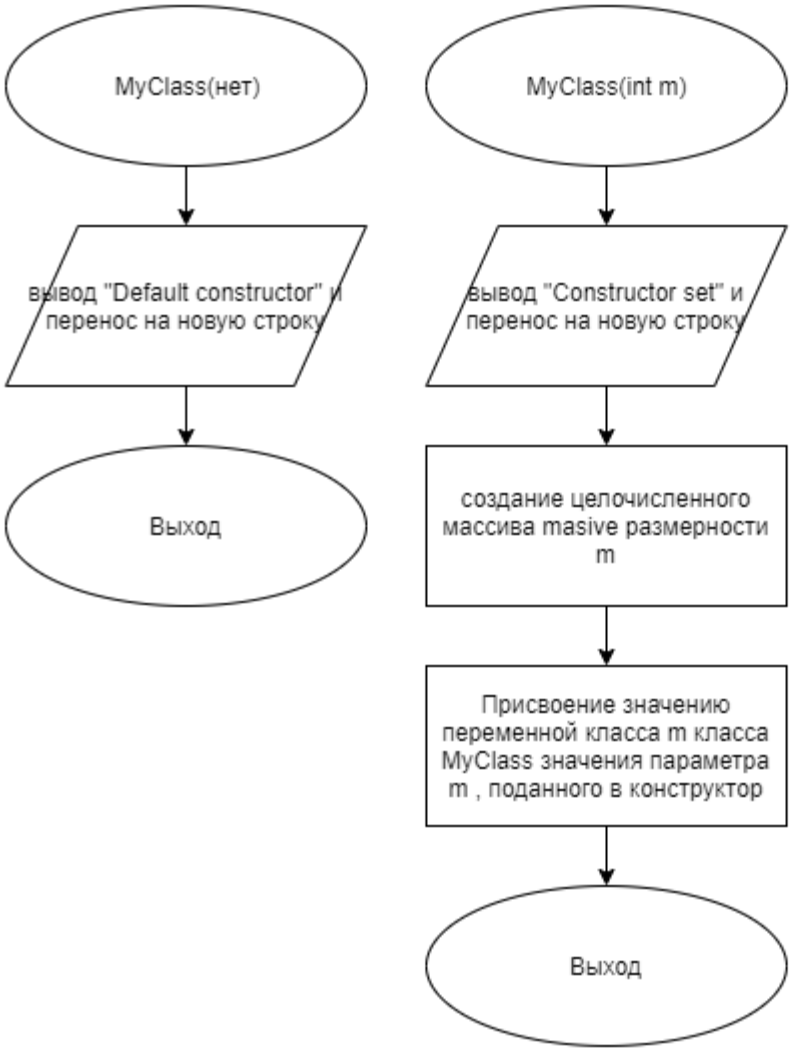

Рисунок 1 – Блок-схема алгоритма

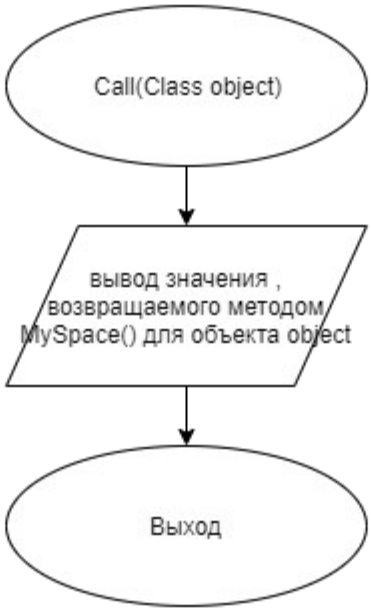

Рисунок 2 – Блок-схема алгоритма

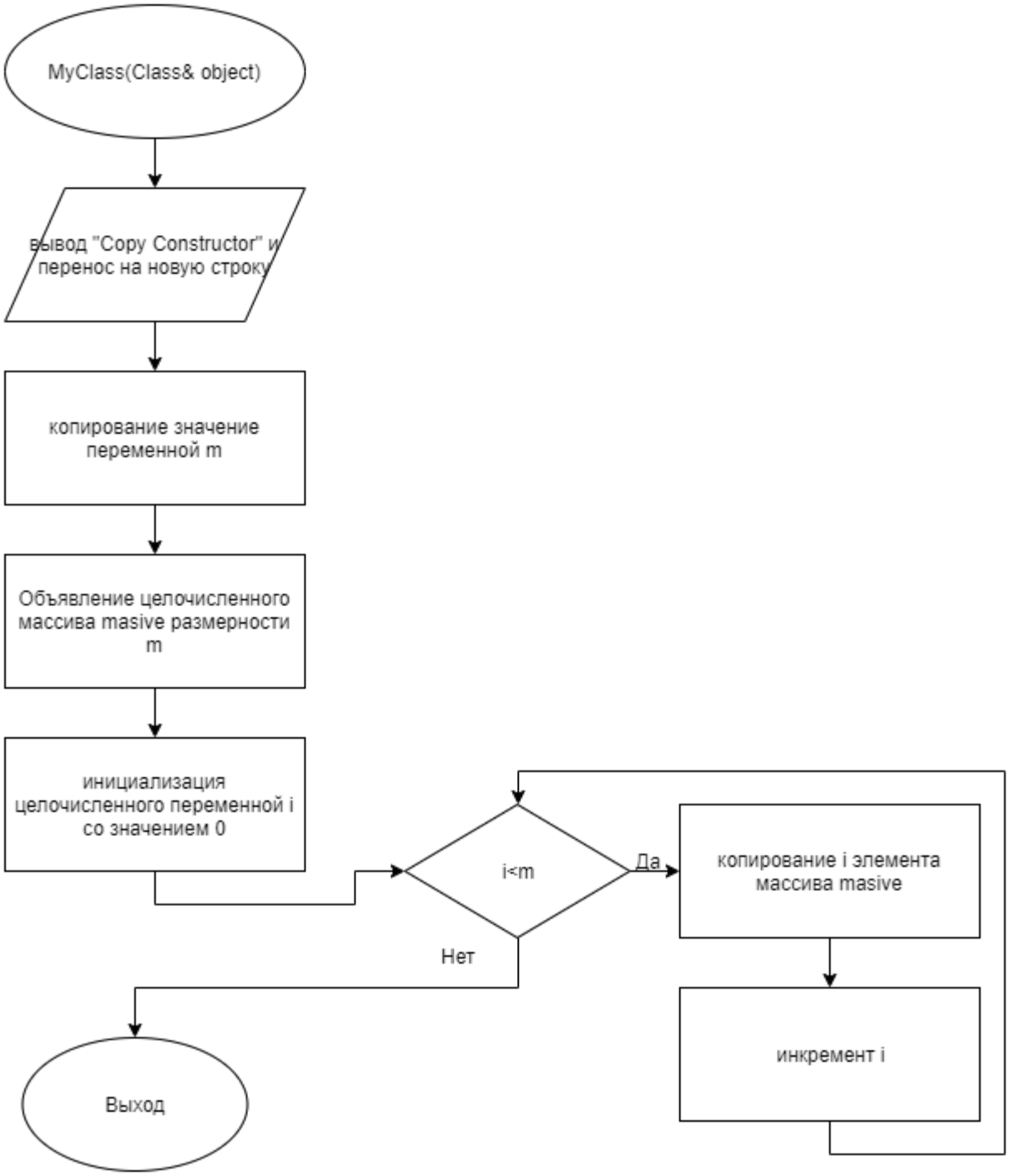

Рисунок 3 – Блок-схема алгоритма

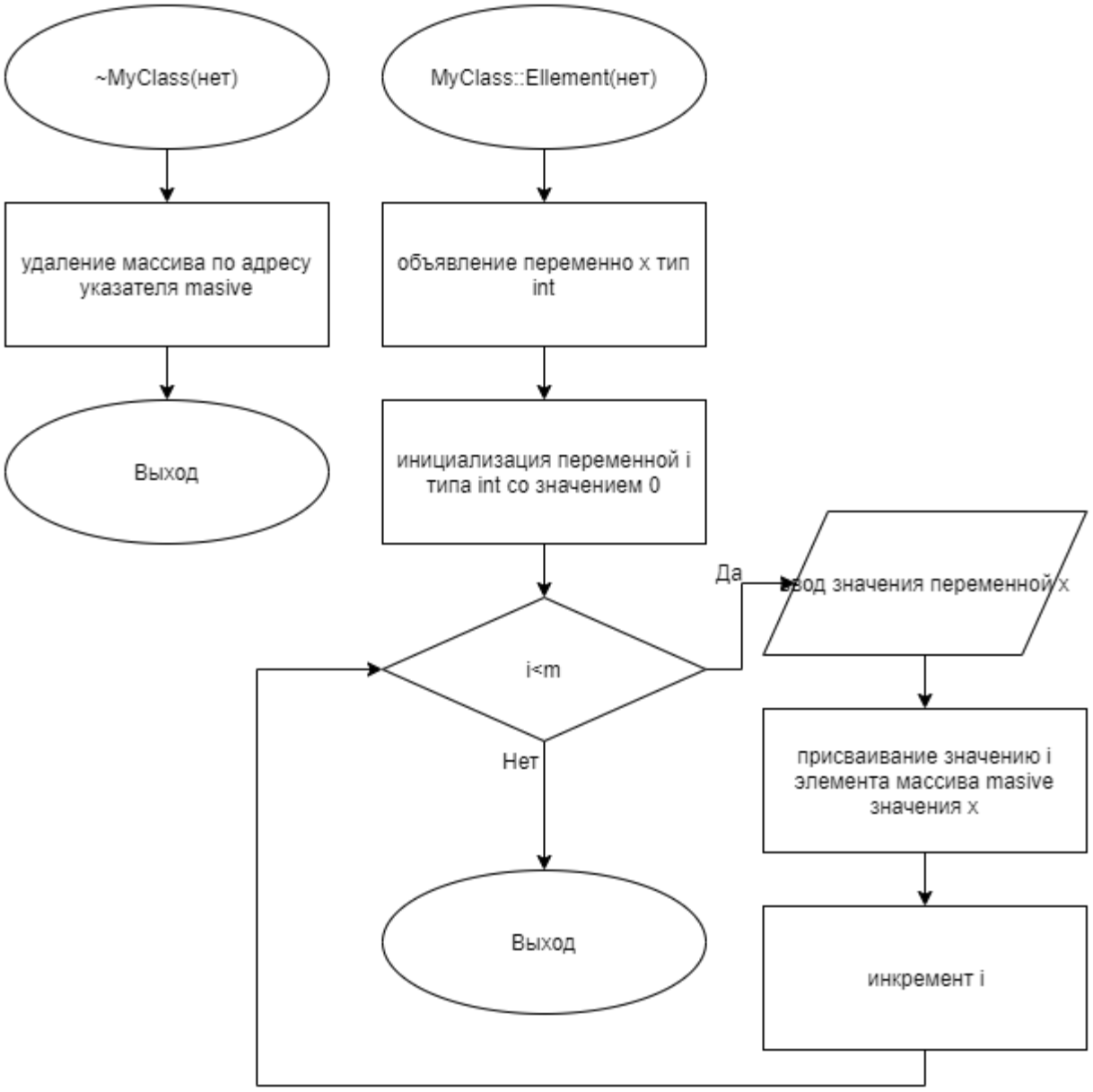

Рисунок 4 – Блок-схема алгоритма

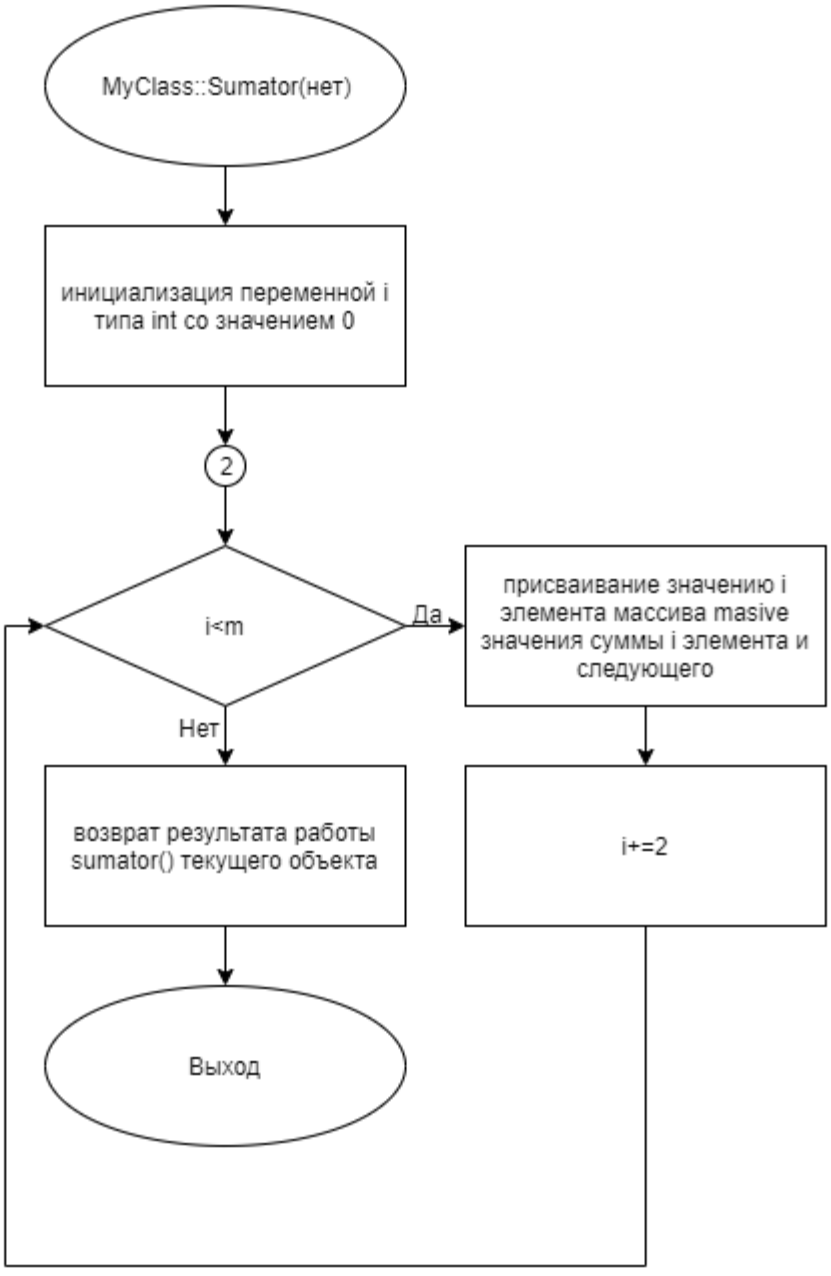

Рисунок 5 – Блок-схема алгоритма

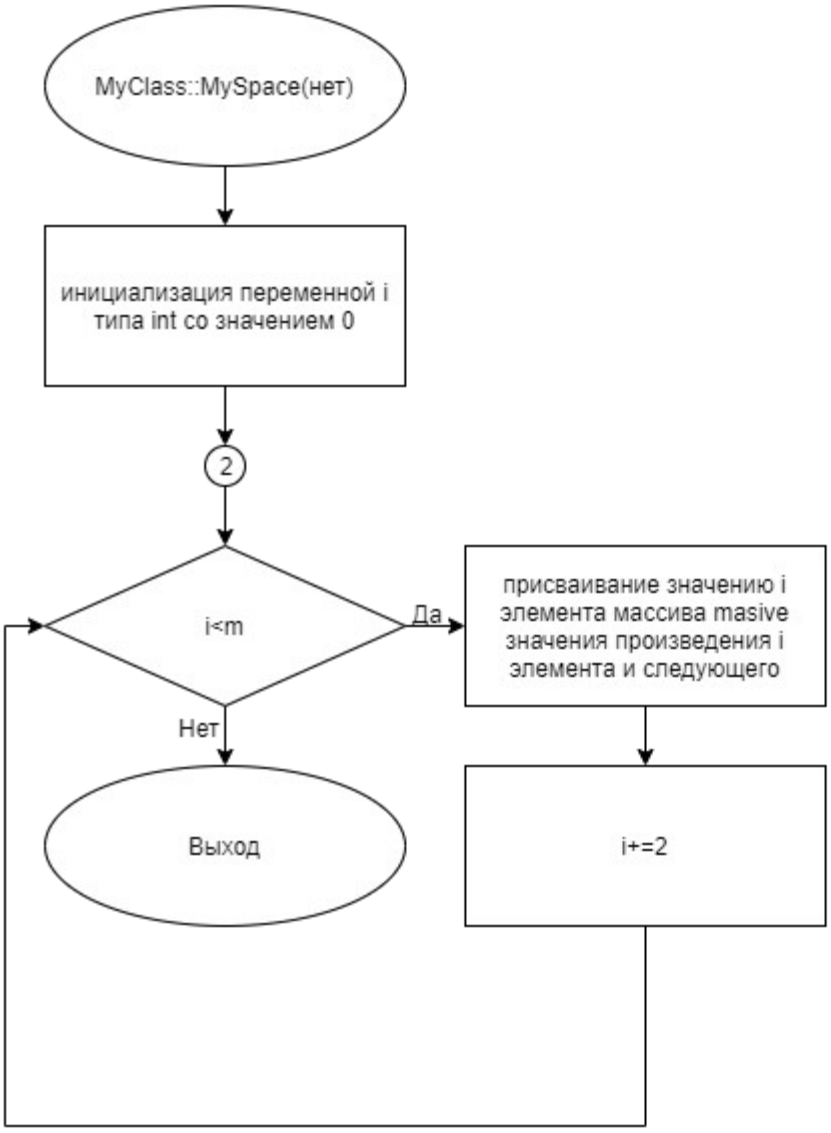

Рисунок 6 – Блок-схема алгоритма

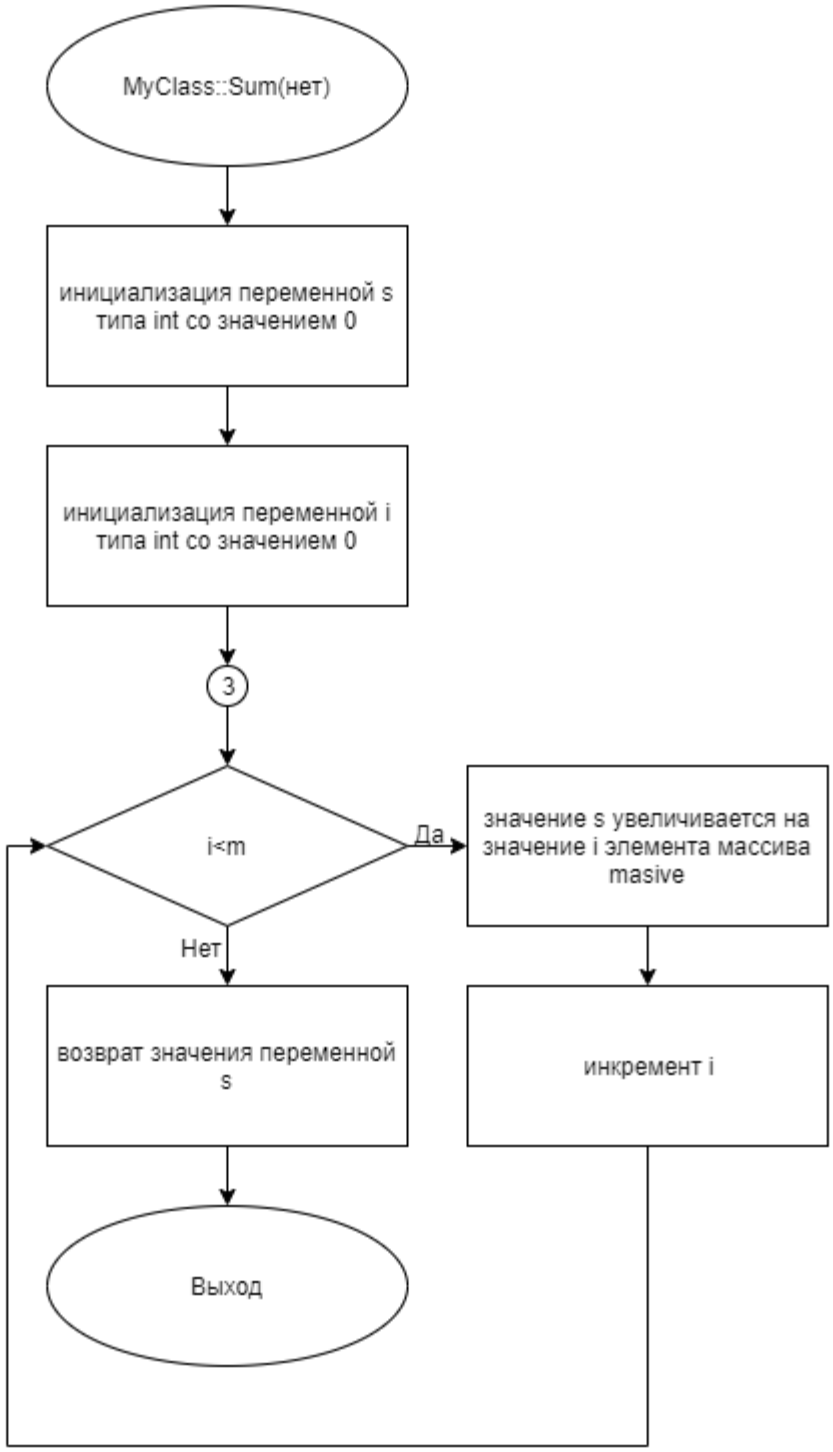

Рисунок 7 – Блок-схема алгоритма

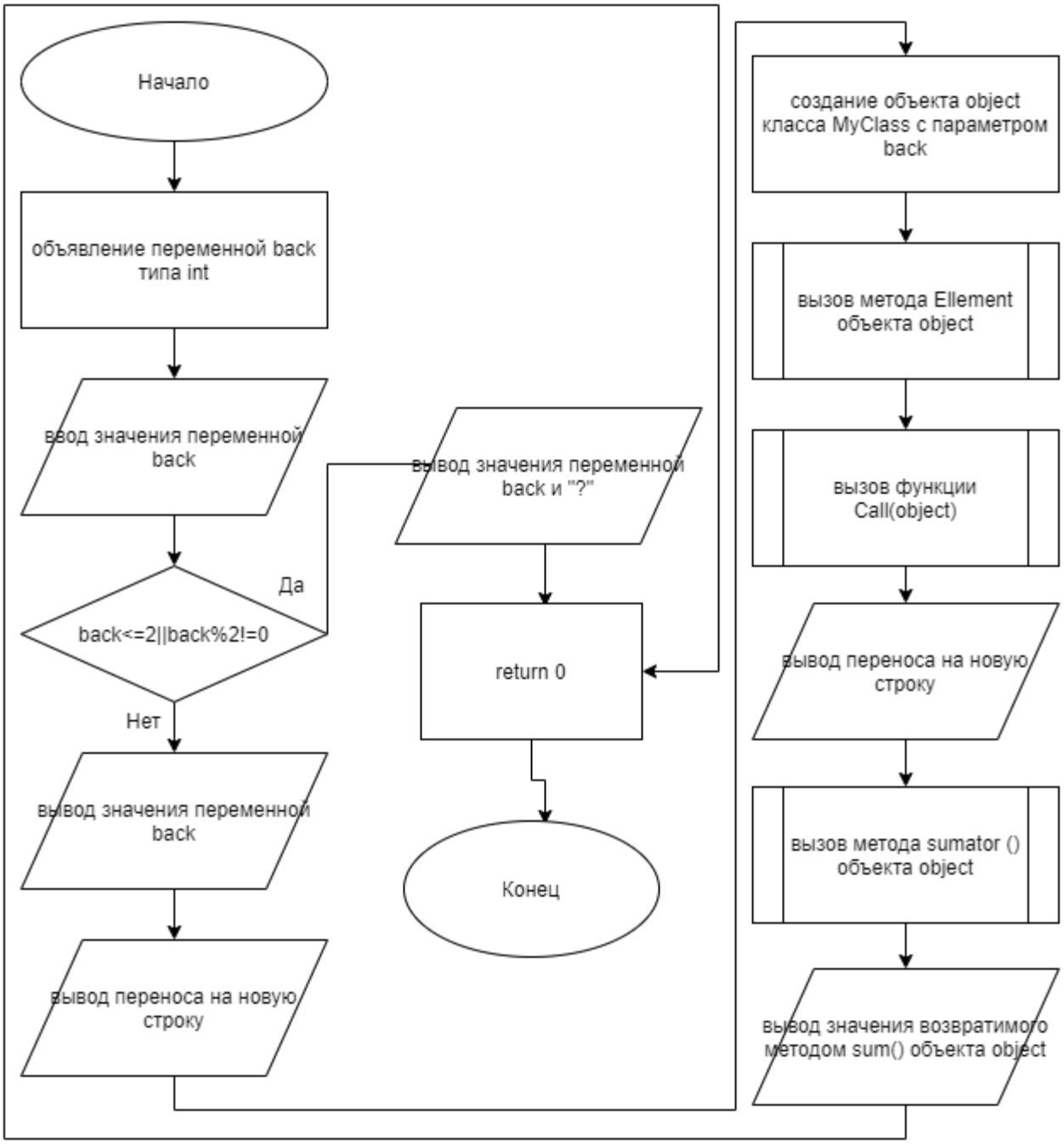

Рисунок 8 – Блок-схема алгоритма

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 "MyClass.h"

using namespace std;

void Call(MyClass ob)
{
   cout<<ob.MySpace();
}


int main()
{
   int back;
   cin>>back;
   if (back<=2||back%2!=0)
   {
      cout<<back<<"?";
      return 0;
   }
   cout <<back;
   cout <<endl;
   MyClass object(back);
   object.Ellement();
   Call(object);
   cout<<endl;
   object.Sumator();
   cout<<object.Sum();
   return(0);
}
```

5.2 Файл MyClass.cpp

Листинг 2 – MyClass.cpp

```
#include "MyClass.h"
#include <iostream>

using namespace std;

MyClass::MyClass()
{
   cout<<"Default constructor"<<endl;
}
MyClass::MyClass(int m)
{
   cout<<"Constructor set"<<endl;
   masive=new int[m];
   this->m=m;
}
MyClass::MyClass(const MyClass& object)
{
   cout<<"Copy constructor"<<endl;
   m=object.m;
   masive=new int[m];
   for (int i=0; i<m; i++)
   {
      masive[i]=object.masive[i];
   }
}
MyClass::~MyClass()
{
   cout<<endl<<"Destructor";
   if (masive!=nullptr)
   {
      delete[] masive;
   }
}
void MyClass::Ellement()
{
   int x;
   for (int i=0; i<m; i++)
   {
      cin>>x;
      masive[i]=x;
   }
}
int MyClass::Sumator()
{
   for (int i=0; i<m; i+=2)
   {
      masive[i]=masive[i]+masive[i+1];
   }
   return Sum();
}
```

```
int MyClass::MySpace()
{
   for (int i=0; i<m; i+=2)
   {
      masive[i]=masive[i]*masive[i+1];
   }
   return Sum();
}
int MyClass::Sum()
{
   int s=0;
   for (int i=0; i<m; i++)
   {
      s+=masive[i];
   }
   return s;
}
```

5.3 Файл MyClass.h

Листинг 3 – MyClass.h

```
#ifndef __MYCLASS__H
#define __MYCLASS__H

using namespace std;
class MyClass
{
   private:
      int m;
      int *masive;
   public:
      MyClass();
      MyClass(int m);
      MyClass(const MyClass& object);
      ~MyClass();
      void Ellement();
      int Sumator();
      int MySpace();
      int Sum();
};

#endif
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 13.

Таблица 13 – Результат тестирования программы

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
8 1 2 3 4 5 6 7 8	8 Constructor set Copy constructor 120 Destructor 56 Destructor	8 Constructor set Copy constructor 120 Destructor 56 Destructor
3 1 2 3	3?	3?
5 1 2 3 4 5	5?	5?
1 1	1?	1?

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).