

Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ.....	5
1.1 Описание входных данных.....	6
1.2 Описание выходных данных.....	7
2 МЕТОД РЕШЕНИЯ.....	9
3 ОПИСАНИЕ АЛГОРИТМОВ.....	10
3.1 Алгоритм метода Create класса Class.....	10
3.2 Алгоритм метода SumPara класса Class.....	10
3.3 Алгоритм метода ProizPara класса Class.....	11
3.4 Алгоритм метода Print класса Class.....	11
3.5 Алгоритм конструктора класса Class.....	12
3.6 Алгоритм функции func.....	12
3.7 Алгоритм функции main.....	13
4 БЛОК-СХЕМЫ АЛГОРИТМОВ.....	15
5 КОД ПРОГРАММЫ.....	20
5.1 Файл Class.cpp.....	20
5.2 Файл Class.h.....	21
5.3 Файл main.cpp.....	22
6 ТЕСТИРОВАНИЕ.....	24
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....	25

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;
- Параметризованный 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 данных для 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 тремя пробелами.

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

1. Создание локального объекта с использованием параметризованного конструктора.
2. Возврат созданного локального объекта.

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. Создание первого объекта.
5. Присвоение первому объекту результата работы функции func с аргументом, содержащим значение размерности массива.
6. Для первого объекта вызов метода создания массива.
7. Для первого объекта вызов метода ввода данных массива.
8. Для первого объекта вызов метода 2.
9. Инициализация второго объекта первым объектом.
10. Вызов метода 1 для второго объекта.
11. Вывод содержимого массива первого объекта.
12. Вывод суммы элементов массива первого объекта.
13. Вывод содержимого массива второго объекта.
14. Вывод суммы элементов массива второго объекта.

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

Параметризованный 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
Default constructor
Constructor set
Destructor
Copy constructor
15  5  2  2
24
20  5  4  2
31
Destructor
Destructor
```

2 МЕТОД РЕШЕНИЯ

Для решения задачи используется:

- функция func для создание локального объекта с помощью параметризованного конструктора и его возврата.

Класс Class:

- функционал:
 - о метод Create — создает целочисленный массив в закрытой области согласно ранее заданной размерности;
 - о метод SumPara — суммирует значения очередной пары элементов и сумму присваивает первому элементу пары;
 - о метод ProizPara — умножает значения очередной пары элементов и результат присваивает первому элементу пары;
 - о метод Print — выводит содержимое массива, элементы разделены тремя пробелами;
 - о метод Class(int n) — параметризованный конструктор, принимающий параметр типа int и присваивающий его значение значению закрытого поля объекта.

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 Create класса Class

Функционал: создает целочисленный массив в закрытой области согласно ранее заданной размерности.

Параметры: none.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 1.

Таблица 1 – Алгоритм метода Create класса Class

№	Предикат	Действия	№ перехода
1		создание целочисленного массива mas размерности n	Ø

3.2 Алгоритм метода SumPara класса Class

Функционал: суммирует значения очередной пары элементов и сумму присваивает первому элементу пары.

Параметры: none.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 2.

Таблица 2 – Алгоритм метода SumPara класса Class

№	Предикат	Действия	№ перехода
1		инициализация переменной i типа int со значением	2

№	Предикат	Действия	№ перехода
		0	
2	$i < n$	присвоение значению i -го элемента массива <code>mas</code> значения суммы i -го элемента массива <code>mas</code> и следующего от него элемента массива <code>mas</code>	3
			Ø
3		увеличение значения переменной i на 2	2

3.3 Алгоритм метода `ProizPara` класса `Class`

Функционал: умножает значения очередной пары элементов и результат присваивает первому элементу пары.

Параметры: none.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода `ProizPara` класса `Class`

№	Предикат	Действия	№ перехода
1		инициализация переменной i типа <code>int</code> со значением 0	2
2	$i < n$	присвоение значению i -го элемента массива <code>mas</code> значения произведения i -го элемента массива <code>mas</code> и следующего от него элемента массива <code>mas</code>	3
			Ø
3		увеличение значения переменной i на 2	2

3.4 Алгоритм метода `Print` класса `Class`

Функционал: выводит содержимое массива, элементы разделены тремя пробелами.

Параметры: none.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода Print класса Class

№	Предикат	Действия	№ перехода
1		инициализация переменной i типа int со значением 0	2
2	i<n	вывод значения i-го элемента массива mas	3
			∅
3	i!=(n-1)	вывод трех пробелов	4
			4
4		инкремент i	2

3.5 Алгоритм конструктора класса Class

Функционал: параметризованный конструктор, принимающий параметр типа int и присваивающий его значение значению закрытого поля объекта.

Параметры: int n.

Алгоритм конструктора представлен в таблице 5.

Таблица 5 – Алгоритм конструктора класса Class

№	Предикат	Действия	№ перехода
1		вывод "Constructor set"	2
2		Свойству n присваивается значение параметра n	∅

3.6 Алгоритм функции func

Функционал: создает локальный объект с помощью параметризованного конструктора и возвращает его.

Параметры: int size.

Возвращаемое значение: Class.

Алгоритм функции представлен в таблице 6.

Таблица 6 – Алгоритм функции func

№	Предикат	Действия	№ перехода
1		создание локального объекта loc с параметром size	2
2		возврат объекта loc	Ø

3.7 Алгоритм функции main

Функционал: запуск программы.

Параметры: none.

Возвращаемое значение: код ошибки (int).

Алгоритм функции представлен в таблице 7.

Таблица 7 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		объявление переменной size типа int	2
2		вывод значения переменной size	3
3	size<=2 or size %2!=0	вывод значения переменной size со знаком вопроса	18
			4
4		вывод значения переменной size	5
5		вывод переноса на новую строку	6
6		создание объекта obj1 класса Class	7
7		присвоение объекту obj1 значения работы функции func(size)	8
8		вывод переноса на новую строку	9
9		вызов метода Create объекта obj1	10

№	Предикат	Действия	№ перехода
10		вызов метода Input объекта obj1	11
11		вызов метода ProizPara объекта obj1	12
12		инициализация объекта obj2 класса Class объектом obj1	13
13		вызов метода SumPara объекта obj2	14
14		вызов метода Print объекта obj1	15
15		вывод результата работы метода Sum объекта obj1 и переноса на новую строку	16
16		вызов метода Print объекта obj2	17
17		вывод результата работы метода Sum объекта obj2	18
18		возврат значения 0	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-5.

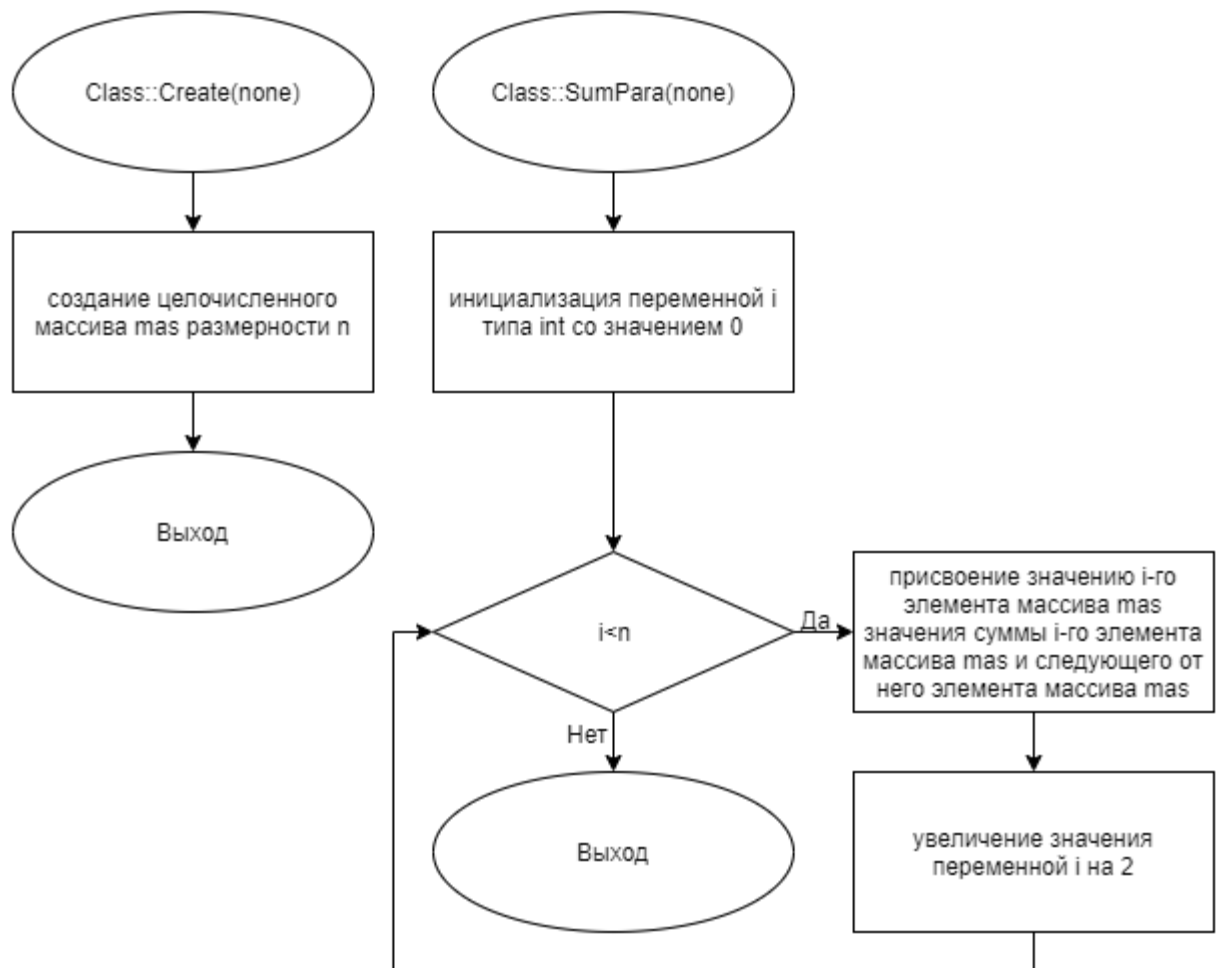


Рисунок 1 – Блок-схема алгоритма

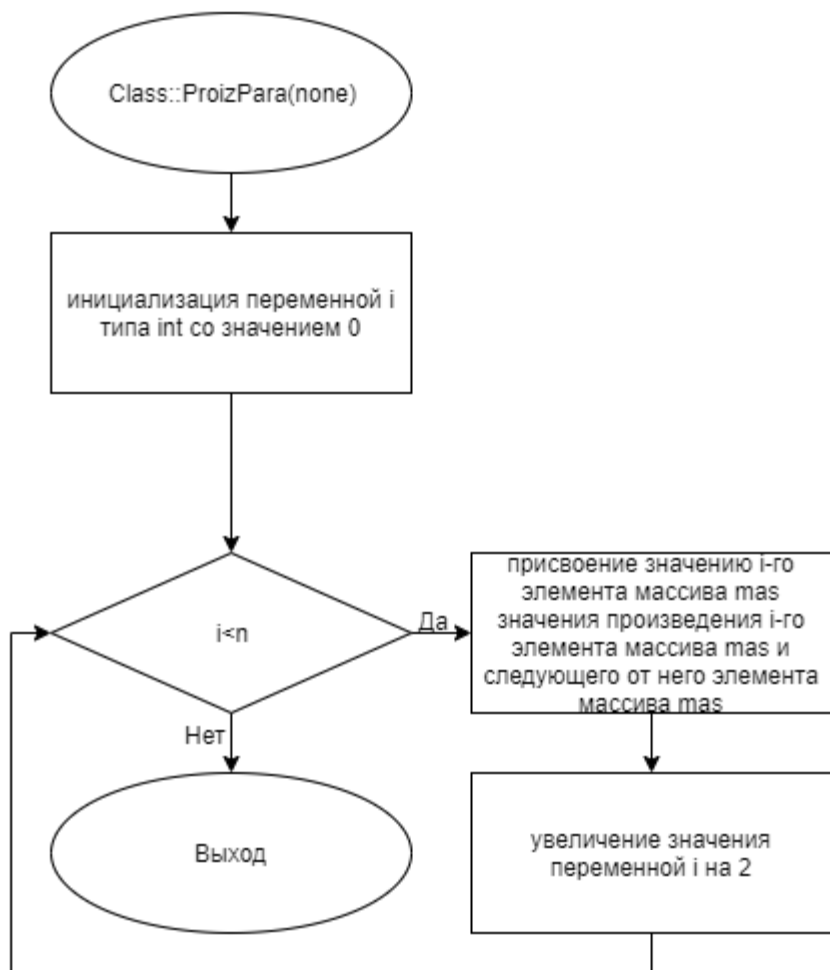


Рисунок 2 – Блок-схема алгоритма

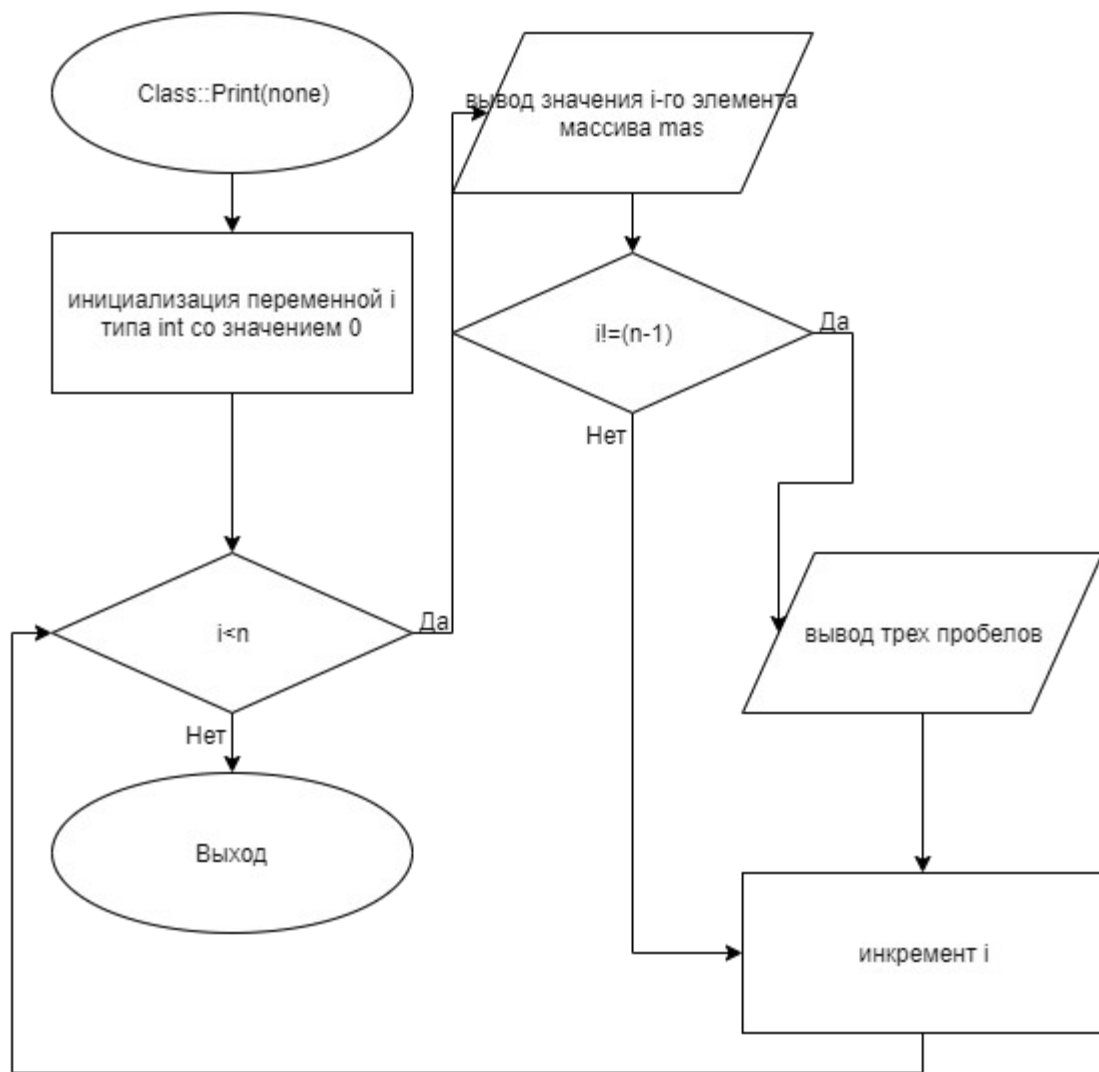


Рисунок 3 – Блок-схема алгоритма

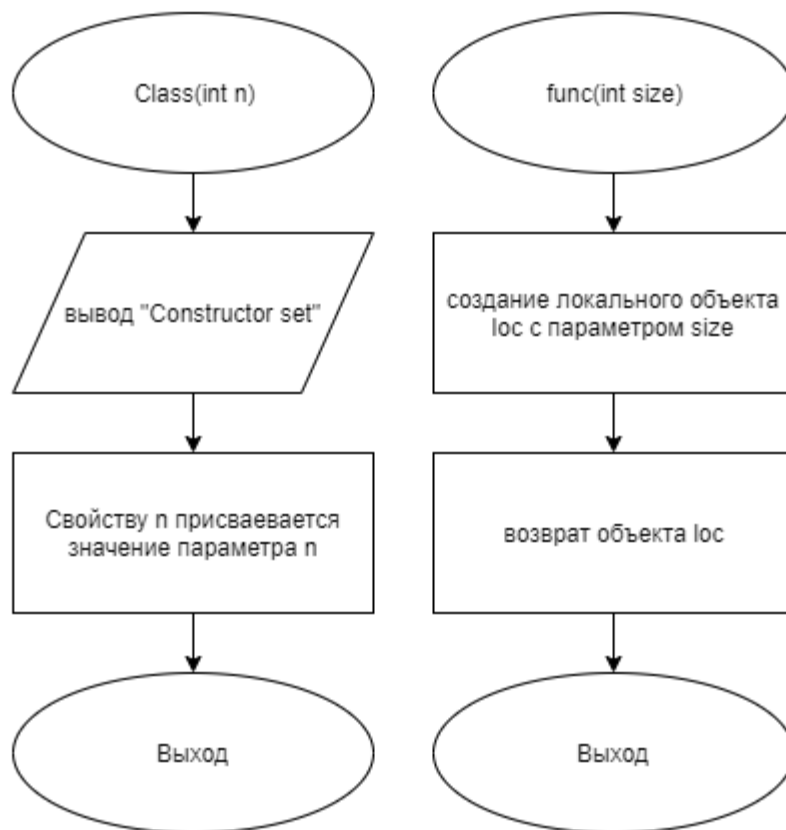


Рисунок 4 – Блок-схема алгоритма

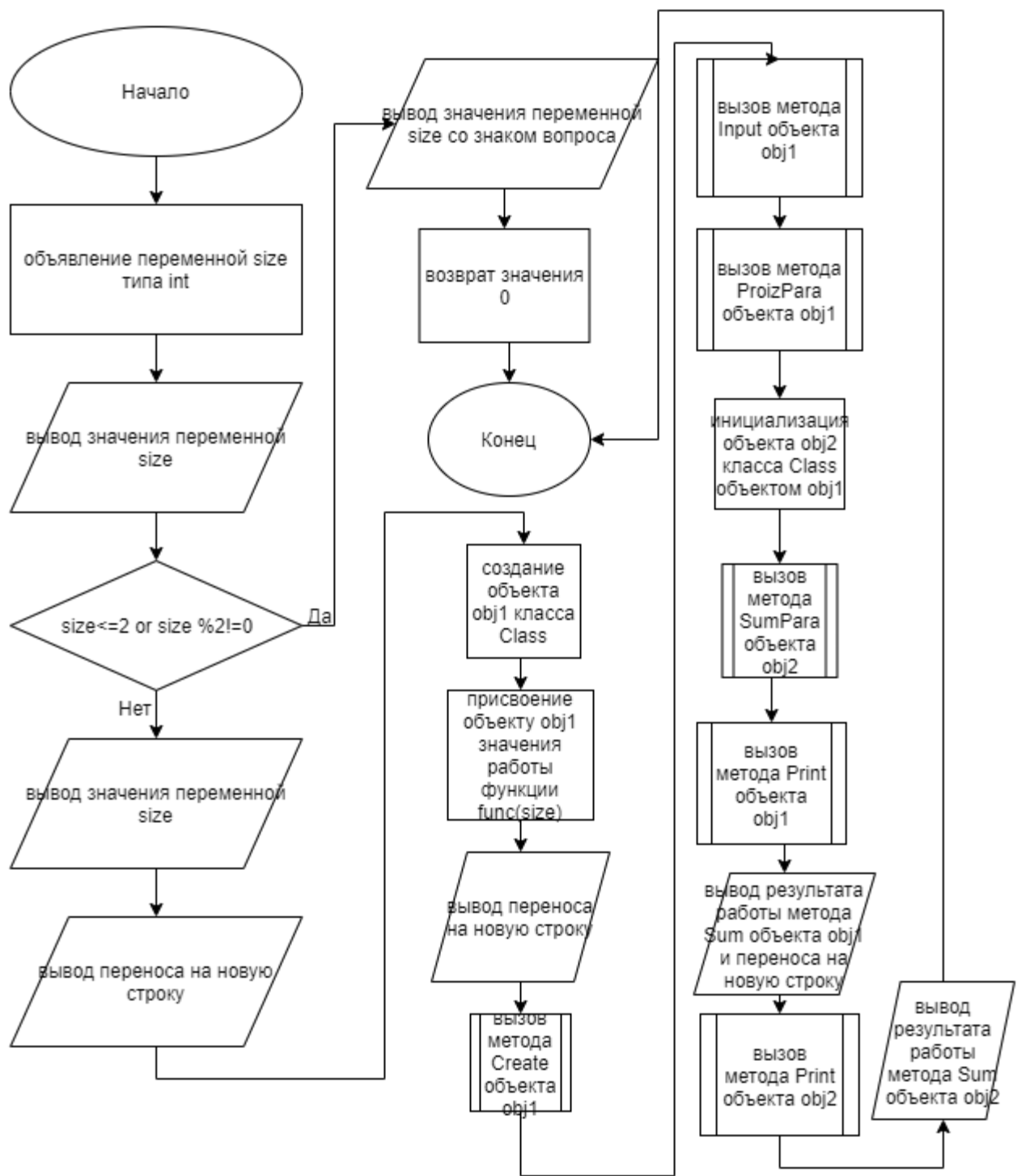


Рисунок 5 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл Class.cpp

Листинг 1 – Class.cpp

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

using namespace std;

Class::Class()
{
    cout<<"Default constructor"<<endl;
}
Class::Class(int n)
{
    cout<<"Constructor set";
    mas=new int[n];
    this->n=n;
}
Class::Class(const Class& obj)
{
    cout<<"Copy constructor"<<endl;
    n=obj.n;
    mas=new int[n];
    for (int i=0; i<n; i++)
    {
        mas[i]=obj.mas[i];
    }
}
Class::~~Class()
{
    cout<<endl<<"Destructor";
    if (mas!=nullptr)
    {
        delete[] mas;
    }
}
void Class::Input()
{
    int x;
    for (int i=0; i<n; i++)
    {
        cin>>x;
```

```

        mas[i]=x;
    }
}
void Class::SumPara()
{
    for (int i=0; i<this->n; i+=2)
    {
        mas[i]=mas[i]+mas[i+1];
    }
}
void Class::ProizPara()
{
    for (int i=0; i<this->n; i+=2)
    {
        mas[i]=mas[i]*mas[i+1];
    }
}
int Class::Sum()
{
    int s=0;
    for (int i=0; i<this->n; i++)
    {
        s+=mas[i];
    }
    return s;
}
void Class::Create()
{
    this->mas=new int[n];
}
void Class::Print()
{
    for (int i=0; i<n; i++)
    {
        cout<<mas[i];
        if (i!=(n-1))
        {
            cout<<" ";
        }
    }
}
}

```

5.2 Файл Class.h

Листинг 2 – Class.h

```

#ifndef __CLASS__H
#define __CLASS__H

using namespace std;

```

```

class Class
{
    private:
        int n;
        int *mas;
    public:
        Class();
        Class(int n);
        Class(const Class& obj);
        ~Class();
        void Create();
        void Input();
        void SumPara();
        void ProizPara();
        int Sum();
        void Print();
};

#endif

```

5.3 Файл main.cpp

Листинг 3 – main.cpp

```

#include <stdlib.h>
#include <stdio.h>
#include <iostream>
#include "Class.h"

using namespace std;

Class func(int size)
{
    Class loc(size);
    return loc;
}

int main()
{
    int size;
    cin>>size;
    if (size<=2||size%2!=0)
    {
        cout<<size<<"?";
        return 0;
    }
    cout<<size;
    cout<<endl;
    Class obj1;
}

```

```
    obj1=func(size);  
    cout<<endl;  
    obj1.Create();  
    obj1.Input();  
    obj1.ProizPara();  
    Class obj2(obj1);  
    obj2.SumPara();  
    obj1.Print();  
    cout<<endl<<obj1.Sum()<<endl;  
    obj2.Print();  
    cout<<endl<<obj2.Sum();  
  
    return 0;  
}
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 8.

Таблица 8 – Результат тестирования программы

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
4 3 5 1 2	4 Default constructor Constructor set Destructor Copy constructor 15 5 2 2 24 20 5 4 2 31 Destructor Destructor	4 Default constructor Constructor set Destructor Copy constructor 15 5 2 2 24 20 5 4 2 31 Destructor Destructor

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-avvora.ru/student/files/methodichescoc_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-avvora.ru/student/files/Prilozheniye_k_methodichke.pdf (дата обращения 05.05.2021).
4. Шилдт Г. С++: 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).