

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.....	8
3 ОПИСАНИЕ АЛГОРИТМОВ.....	10
3.1 Алгоритм конструктора класса Class.....	10
3.2 Алгоритм метода Method1 класса Class.....	10
3.3 Алгоритм метода Method2 класса Class.....	11
3.4 Алгоритм деструктора класса Class.....	11
3.5 Алгоритм конструктора класса Class.....	12
3.6 Алгоритм конструктора класса Class.....	12
3.7 Алгоритм функции Func.....	13
3.8 Алгоритм функции main.....	13
3.9 Алгоритм метода Sum класса Class.....	14
3.10 Алгоритм метода Input класса Class.....	14
4 БЛОК-СХЕМЫ АЛГОРИТМОВ.....	16
5 КОД ПРОГРАММЫ.....	23
5.1 Файл Class.cpp.....	23
5.2 Файл Class.h.....	24
5.3 Файл main.cpp.....	24
6 ТЕСТИРОВАНИЕ.....	26
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....	27

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;
- Параметризованный конструктор, передается целочисленный параметр. Параметр должен иметь значение больше 2 и быть четным. По значению параметра определяется размерность целочисленного массива из закрытой области. 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.
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 а класса Class предназначен для работа с целочисленным массивом;
- функция new для Выделение памяти под объект;
- функция delete для Освобождение выделенной памяти;
- cin - стандартный объект потока ввода данных;
- cout - стандартный объект потока вывода данных;
- for - оператор начала цикла;
- if - условный оператор.

Класс Class:

- свойства/поля:
 - поле Хранение длины массива:
 - наименование — n;
 - тип — int;
 - модификатор доступа — private;
 - поле Указатель на целочисленный массив:
 - наименование — mas;
 - тип — int *;
 - модификатор доступа — private;
- функционал:
 - метод Class — Конструктор по умолчанию;
 - метод Class — Параметризированный конструктор;
 - метод Class — Конструктор копии объекта;
 - метод Sum — Возвращение суммы элементов массива mas;
 - метод Input — Ввод значений элементов mas;

- o метод Method1 — Операции сложения над элементами массива;
- o метод Method2 — Операции умножения над элементами массива;
- o метод ~Class — Деструктор.

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 Class

Функционал: Конструктор по умолчанию.

Параметры: нет.

Алгоритм конструктора представлен в таблице 1.

Таблица 1 – Алгоритм конструктора класса Class

№	Предикат	Действия	№ перехода
1		Вывод с новой строки слов Default constructor	Ø

3.2 Алгоритм метода Method1 класса Class

Функционал: Присваивает нечетным элементам массива значения сумм их со следующими элементами.

Параметры: нет.

Возвращаемое значение: int - сумма элементов массива mas.

Алгоритм метода представлен в таблице 2.

Таблица 2 – Алгоритм метода Method1 класса Class

№	Предикат	Действия	№ перехода
1		инициализация целочисленной s = 0	2
2		инициализация переменной-счетчика i=0	3
3	i < n	присвоение mas[i] = mas[i] + mas[i+1]; присвоение	3

№	Предикат	Действия	№ перехода
		s = mas[i] + mas[i+1]; присвоить i значение i+2	
			4
4		возвращение значения s	∅

3.3 Алгоритм метода Method2 класса Class

Функционал: Присваивает нечетным элементам массива значения их произведения на следующий элемент.

Параметры: нет.

Возвращаемое значение: int - сумма элементов массива mas.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода Method2 класса Class

№	Предикат	Действия	№ перехода
1		инициализация целочисленной s = 0	2
2		инициализация переменной-счетчика i=0	3
3	i<n	присвоение mas[i] = mas[i] * mas[i+1]; присвоение s = mas[i] + mas[i+1]; присвоить i значение i+2	3
			4
4		возвращение значения s	∅

3.4 Алгоритм деструктора класса Class

Функционал: Уничтожение объекта.

Параметры: нет.

Алгоритм деструктора представлен в таблице 4.

Таблица 4 – Алгоритм деструктора класса Class

№	Предикат	Действия	№ перехода
1		Вывод с новой строки слова Destructor	2
2		Освобождение выделенной под mas памяти	∅

3.5 Алгоритм конструктора класса Class

Функционал: Создание объектов.

Параметры: целочисленное, n1, длина массива mas .

Алгоритм конструктора представлен в таблице 5.

Таблица 5 – Алгоритм конструктора класса Class

№	Предикат	Действия	№ перехода
1		вывод с новой строки слов Constructor set	2
2		присвоение n значения n1	3
3		выделение памяти для mas с n количеством элементов	∅

3.6 Алгоритм конструктора класса Class

Функционал: Конструктор копии.

Параметры: const Class & a - объекты для присвоения полям значений.

Алгоритм конструктора представлен в таблице 6.

Таблица 6 – Алгоритм конструктора класса Class

№	Предикат	Действия	№ перехода
1		вывод с новой строки слов Copy constructor	2
2		присвоение n значения a.n из объекта a класса Class	3
3		выделение памяти для копии массива mas	4
4		присвоение значениям элементов mas значений элементов a.mas объекта a класса Class	∅

3.7 Алгоритм функции Func

Функционал: выполнение Method2 для копии объекта.

Параметры: объекта a_local класса Class.

Возвращаемое значение: void - не возвращает значения.

Алгоритм функции представлен в таблице 7.

Таблица 7 – Алгоритм функции Func

№	Предикат	Действия	№ перехода
1		обращение к методу Method2() для объекта a.local класса Class	2
2		обращение к методу Sum() для объекта a.local класса Class	Ø

3.8 Алгоритм функции main

Функционал: основной алгоритм программы.

Параметры: нет.

Возвращаемое значение: целочисленное - индикатор корректности завершения программы.

Алгоритм функции представлен в таблице 8.

Таблица 8 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		объявление целочисленной n1	2
2		ввод n1 с клавиатуры	3
3	n1 <=2 n1%2!=0	вывод с новой строки n1 "?"	Ø
		вывод с новой строки n1	4
4		создание объекта a класса Class с параметром n1	5
5		вызов метода Input для объекта a	6
6		вызов функции Func с параметром объект a	7
7		вызов метода Method1 для объекта a	8

№	Предикат	Действия	№ перехода
8		вывод с новой строки значения возвращаемого методом Sum для объекта a	Ø

3.9 Алгоритм метода Sum класса Class

Функционал: Возвращение суммы элементов массива mas.

Параметры: нет.

Возвращаемое значение: int - сумма элементов массива mas.

Алгоритм метода представлен в таблице 9.

Таблица 9 – Алгоритм метода Sum класса Class

№	Предикат	Действия	№ перехода
1		инициализация целочисленной s=0	2
2		инициализация переменной счетчика i=0	3
3	i<n	присвоение s значения s+mas[i]; инкремент i	3
			4
4		возвращение значения s	Ø

3.10 Алгоритм метода Input класса Class

Функционал: Ввод значений элементов mas.

Параметры: нет.

Возвращаемое значение: void - не возвращает значения.

Алгоритм метода представлен в таблице 10.

Таблица 10 – Алгоритм метода Input класса Class

№	Предикат	Действия	№ перехода
1		инициализация переменной счетчика i=0	2
2	i<n	ввод с клавиатуры значения mas[i]; инкремент i	2

№	Предикат	Действия	№ перехода
			Ø

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-7.

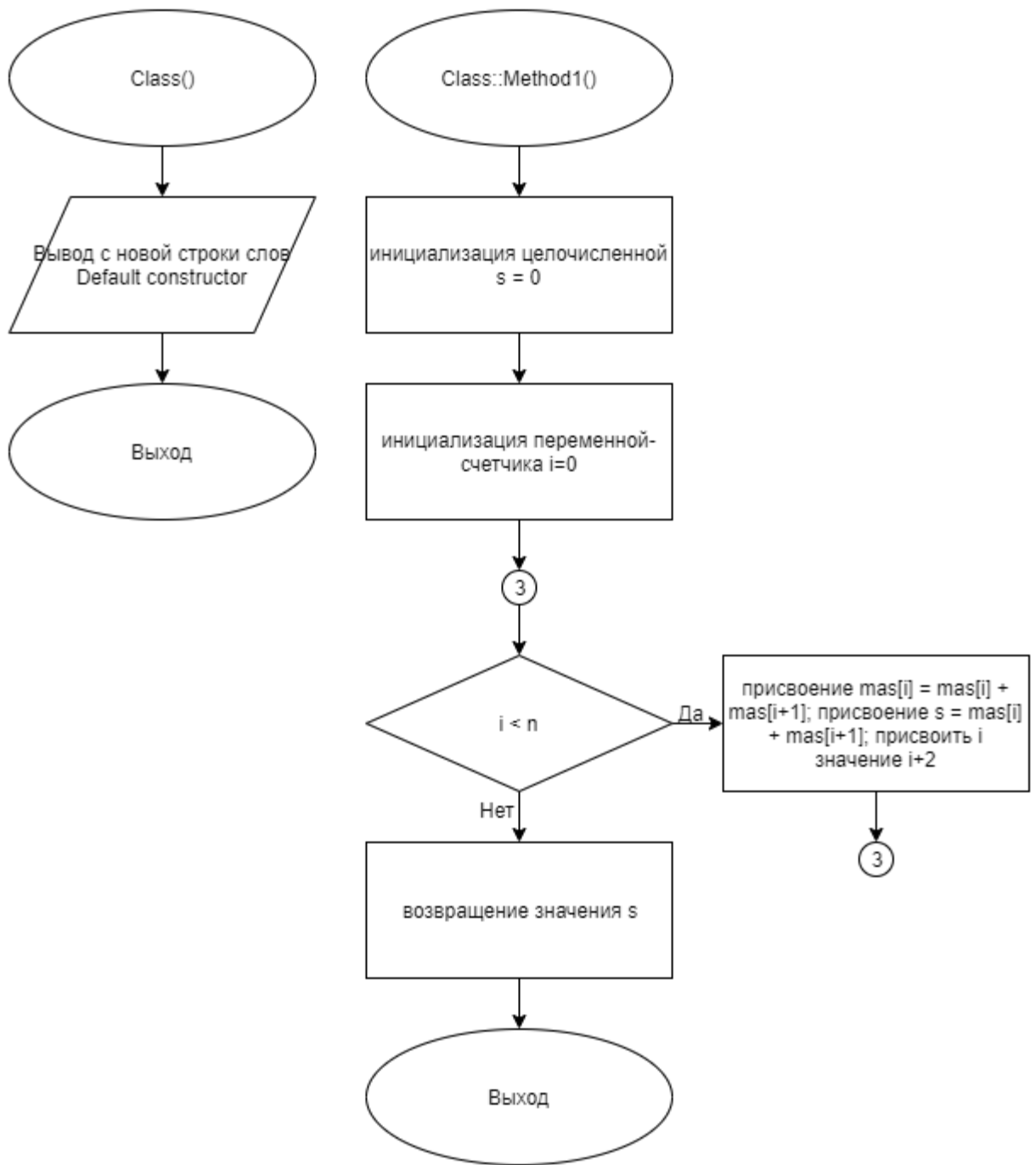


Рисунок 1 – Блок-схема алгоритма

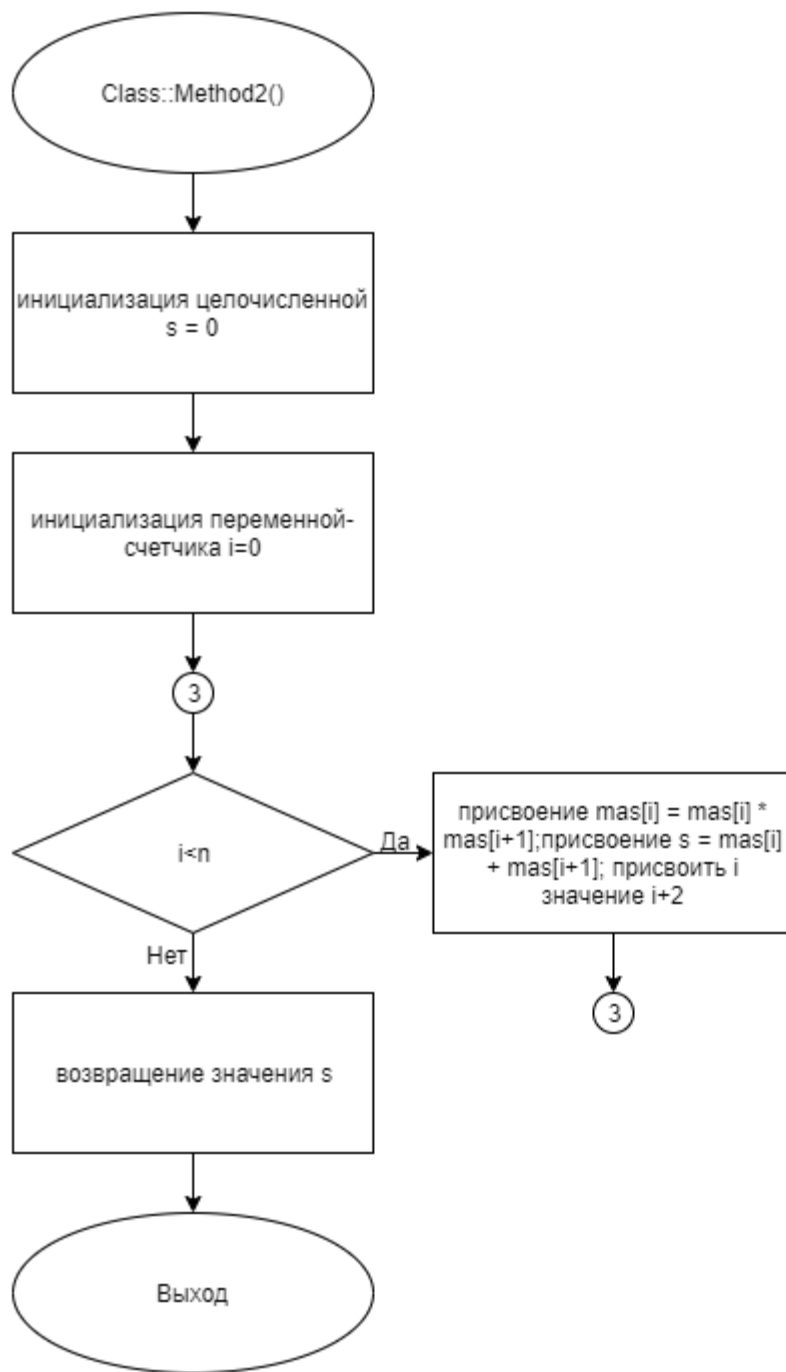


Рисунок 2 – Блок-схема алгоритма

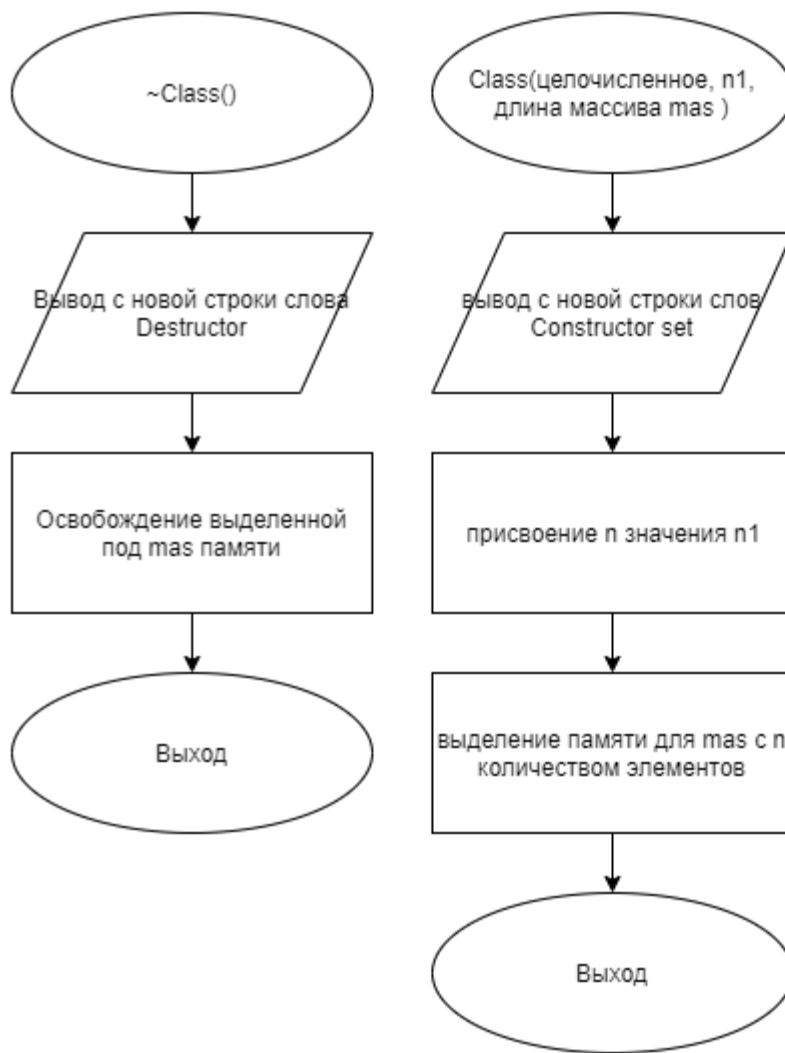


Рисунок 3 – Блок-схема алгоритма

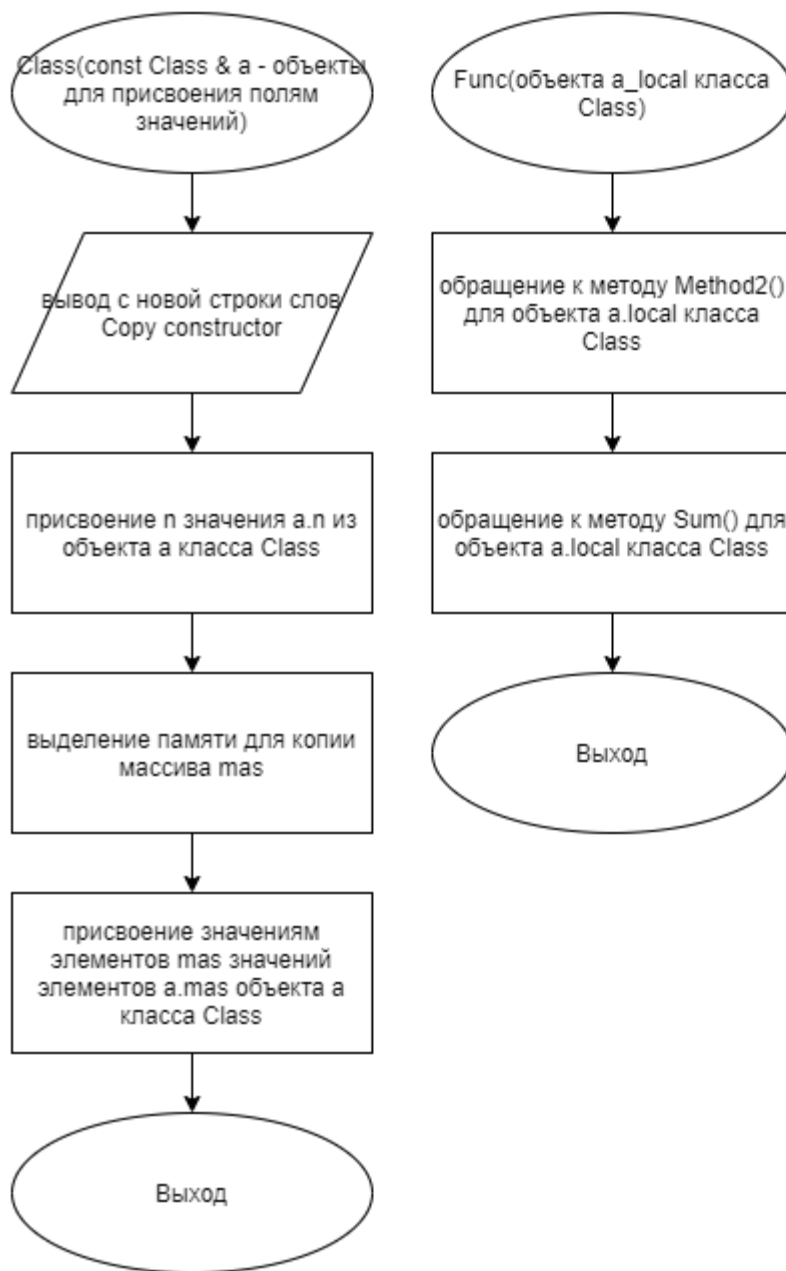


Рисунок 4 – Блок-схема алгоритма

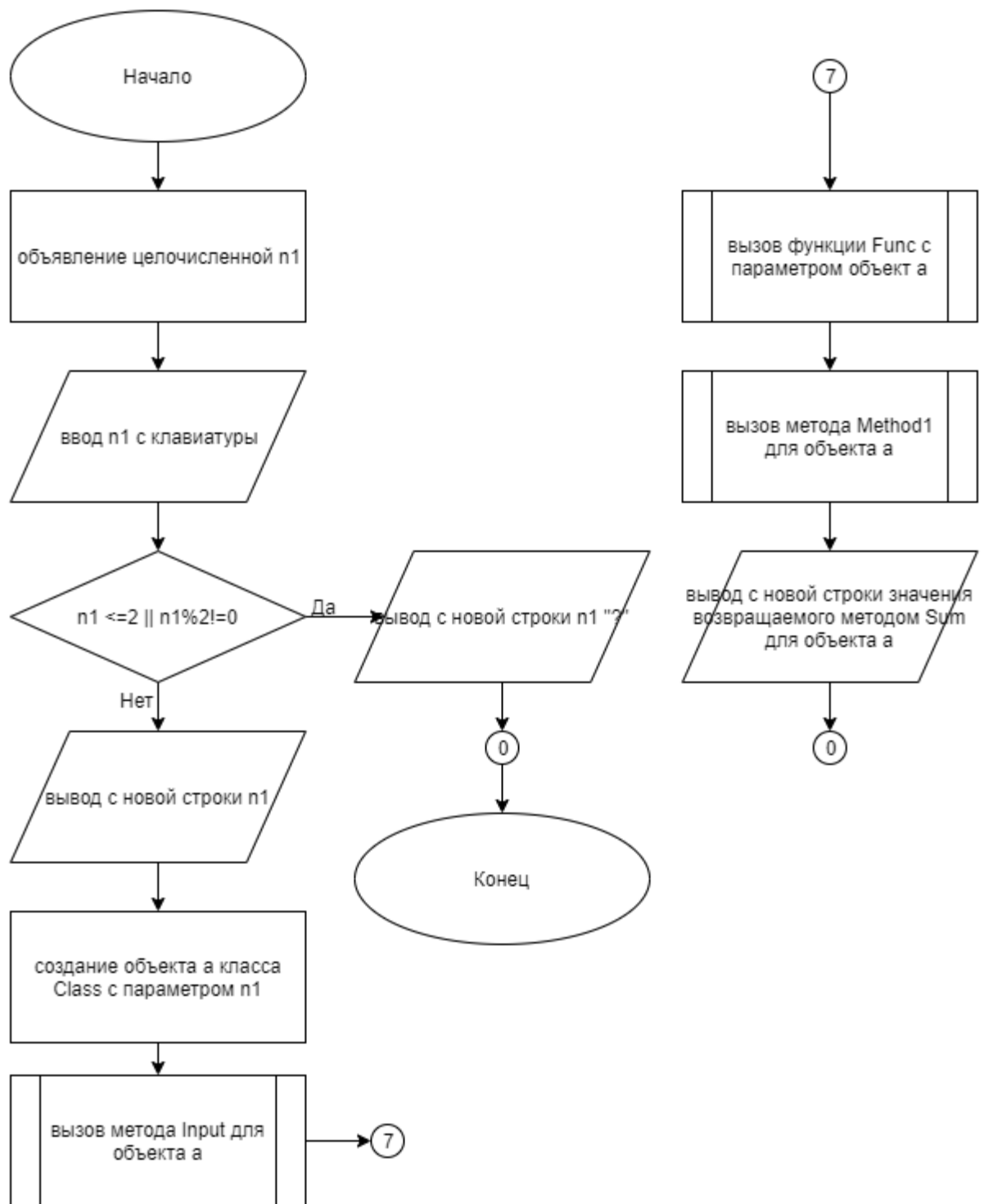


Рисунок 5 – Блок-схема алгоритма

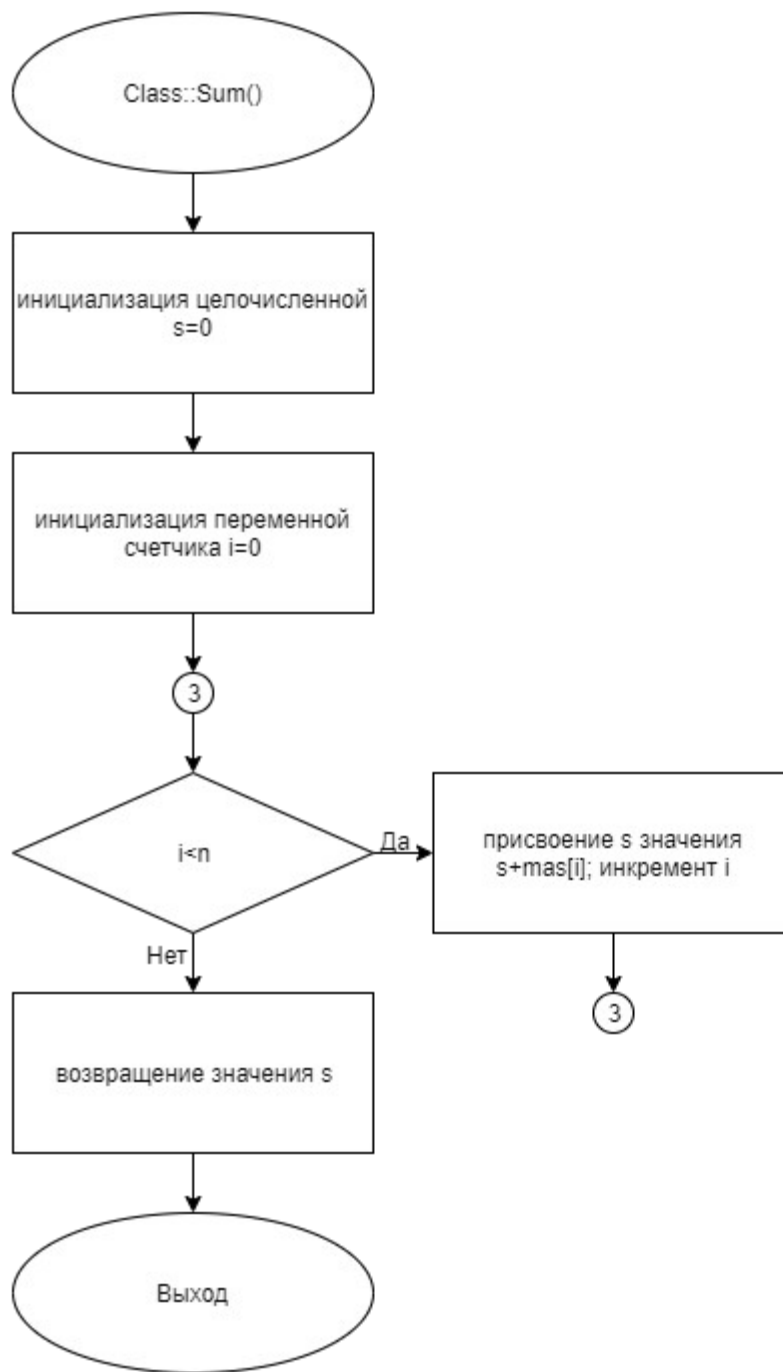


Рисунок 6 – Блок-схема алгоритма

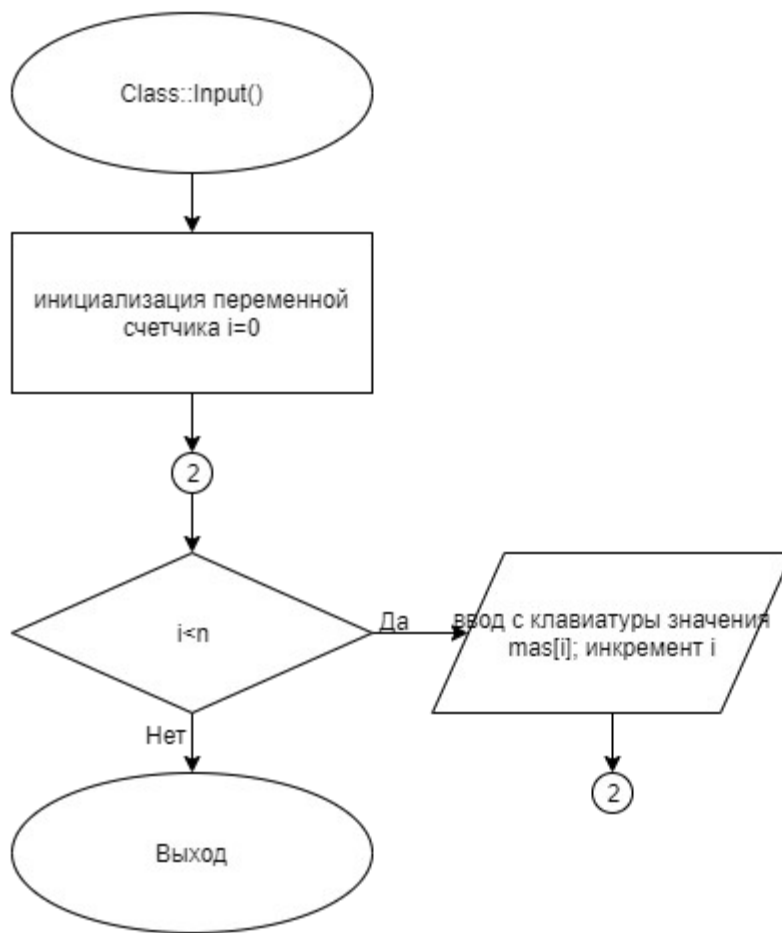


Рисунок 7 – Блок-схема алгоритма

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 n1){
    cout<<"Constructor set"<<endl;
    n=n1;
    mas=new int[n];
}

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

void Class::Input(){
    for (int i=0;i<n;i++){
        cin>>mas[i];
    }
}

int Class::Method1(){
    int s=0;
    for (int i=0;i<n;i+=2){
        mas[i]=mas[i]+mas[i+1];
        s+=mas[i]+mas[i+1];
    }
    return s;
}

int Class::Method2(){
    int s=0;
    for (int i=0;i<n;i+=2){
        mas[i]=mas[i]*mas[i+1];
        s+=mas[i]+mas[i+1];
    }
}
```

```

    }
    return s;
}
int Class::Sum(){
    int sum=0;
    for (int i=0;i<n;i++){
        sum+=mas[i];
    }
    return sum;
}

Class::~~Class(){
    cout<<"Destructor"<<endl;
    delete[] mas;
}

```

5.2 Файл Class.h

Листинг 2 – Class.h

```

#ifndef __CLASS__H
#define __CLASS__H
class Class{
    private:
        int* mas;
        int n;
    public:
        Class();
        Class(int n1);
        Class(const Class & a);
        void Input();
        int Method1();
        int Method2();
        int Sum();
        ~Class();

};

#endif

```

5.3 Файл main.cpp

Листинг 3 – main.cpp

```

#include <stdlib.h>

```

```

#include <stdio.h>
#include "Class.h"
#include <iostream>
using namespace std;
void Func(Class a_local){
    a_local.Method2();
    cout<<a_local.Sum()<<endl;
}
int main()
{
    int n1;
    cin>>n1;
    if (n1<=2 || n1%2!=0){
        cout<<n1<<"?";
        return(0);
    }
    cout<<n1<<endl;
    Class a(n1);
    a.Input();
    Func(a);
    a.Method1();
    cout<<a.Sum()<<endl;
    return(0);
}

```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 11.

Таблица 11 – 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

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/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-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).