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. 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:

- Ввод размерности массива.
- Вывод значения размерности массива.
- Если размерность массива некорректная, вывод сообщения и завершить работу алгоритма.
- Вывод значения размерности массива.
- Создание объекта с аргументом размерности массива.
- Вызов метода для ввода значений элементов массива.
- Вызов функции передача в качестве аргумента объекта.
- Вызов метода 1 от имени объекта.
- 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

Операторы потока ввода/вывода cin, cout;

Оператор функция new;

Оператор функция delete;

Объект obj класса Class;

Функция Call(Class o) - вывод результата работы метода Prod() объекта o;

Класс Class:

- Свойства/поля:
  - Поле размера массива mas:
    - Наименование - n;
    - Тип - int;
    - Модификатор доступа - закрытый;
  - Поле целочисленного массива mas:
    - Наименование - mas;
    - Тип - int*;
    - Модификатор доступа - закрытый;
- Функционал:
  - Class() - конструктор по умолчанию, в начале работы выдает сообщение об отработке;
  - Class(int n) - параметризированный 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 об отработке;
  - Class(const Class& obj) - конструктор копии для исключения ошибки при передаче объекта в функцию по значению;

- ~Class() - деструктор, выдаёт сообщение об отработке;
- Input() - метод для ввода значений элементов массива;
- SumPara() - метод: суммирует значения очередной пары элементов и сумму присваивает первому элементу пары, далее суммирует элементы полученного массива и возвращает это значение;
- Prod() - метод: умножает значения очередной пары элементов и результат присваивает первому элементу пары, далее суммирует элементы полученного массива и возвращает это значение;
- 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 Class

Функционал: Вывод сообщения об отработке.

Параметры: .

Алгоритм конструктора представлен в таблице 1.

Таблица 1 – Алгоритм конструктора класса Class

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

3.2 Алгоритм конструктора класса Class

Функционал: передается целочисленный параметр, по значению параметра определяется размерность целочисленного массива из закрытой области, начале работы выдает сообщение.

Параметры: int n.

Алгоритм конструктора представлен в таблице 2.

Таблица 2 – Алгоритм конструктора класса Class

№	Предикат	Действия	№ перехода
1		Вывод "Constructor set" и перенос на новую строку	2
2		Создание целочисленного массива mas размерности n	3
3		Присвоение значению переменной n класса Class значение параметра	∅

№	Предикат	Действия	№ перехода
		n, поданного в конструктор	

3.3 Алгоритм конструктора класса Class

Функционал: создание объекта в качестве копии объекта, подаваемого через параметр.

Параметры: const Class& obj.

Алгоритм конструктора представлен в таблице 3.

Таблица 3 – Алгоритм конструктора класса Class

№	Предикат	Действия	№ перехода
1		Вывод "Copy Constructor" и перенос на новую строку	2
2		копирование значения переменной n	3
3		Объявление целочисленного массива mas размерности n	4
4		Инициализация целочисленной переменной i значением 0	5
5	i < n	копирование i-того элемента массива mas	6
			∅
6		i += 1	5

3.4 Алгоритм деструктора класса Class

Функционал: вывод сообщения об отработке.

Параметры: .

Алгоритм деструктора представлен в таблице 4.

Таблица 4 – Алгоритм деструктора класса Class

№	Предикат	Действия	№ перехода
1		Удаление массива по адресу указателя mas, созданного при помощи оператора new	∅

3.5 Алгоритм метода Input класса Class

Функционал: ввод значений элементов массива.

Параметры: .

Возвращаемое значение: void.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода Input класса Class

№	Предикат	Действия	№ перехода
1		Объявление целочисленной переменной x	2
2		Инициализация целочисленной переменной i значением 0	3
3	i < n	Ввод значения переменной x	4
			∅
4		mas[i] = x	5
5		i += 1	∅

3.6 Алгоритм метода SumPara класса Class

Функционал: суммирует значения очередной пары элементов и сумму присваивает первому элементу пары, далее суммирует элементы полученного массива и возвращает это значение.

Параметры: .

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 6.

Таблица 6 – Алгоритм метода SumPara класса Class

№	Предикат	Действия	№ перехода
1		Инициализация целочисленной переменной i значением 0	2
2	i < n	mas[i] = mas[i] + mas[i + 1]	3
		Возврат результата работы метода Sum() текущего объекта	∅
3		i += 2	2

3.7 Алгоритм метода Prod класса Class

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

Параметры: .

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода Prod класса Class

№	Предикат	Действия	№ перехода
1		Инициализация целочисленной переменной i значением 0	2
2	i < n	mas[i] = mas[i] * mas[i + 1]	3
		Возврат результата работы метода Sum() текущего объекта	∅
3		i += 2	2

3.8 Алгоритм метода Sum класса Class

Функционал: суммирует значения элементов массива.

Параметры: .

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 8.

Таблица 8 – Алгоритм метода Sum класса Class

№	Предикат	Действия	№ перехода
1		Инициализация целочисленной переменной s значением 0	2
2		Инициализация целочисленной переменной i значением 0	3
3	i < n	s += mas[i]	4
		Возврат значения переменной s	∅
4		i += 1	3

3.9 Алгоритм функции Call

Функционал: Вывод результата работы метода Prod объекта o.

Параметры: Class o.

Возвращаемое значение: void.

Алгоритм функции представлен в таблице 9.

Таблица 9 – Алгоритм функции Call

№	Предикат	Действия	№ перехода
1		Вывод значения, возвращаемого методом Prod() для объекта o	∅

3.10 Алгоритм функции main

Функционал: Ввод размерности массива. Вывод значения размерности массива. Если размерность массива некорректная, вывод сообщения и завершить работу алгоритма. Вывод значения размерности массива. Создание объекта с аргументом размерности массива. Вызов метода для ввода значений элементов массива. Вызов функции передача в качестве аргумента объекта. Вызов метода 1 от имени объекта. Вывод суммы элементов массива объекта с новой строки.

Параметры: .

Возвращаемое значение: int - код возврата.

Алгоритм функции представлен в таблице 10.

Таблица 10 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		Объявление целочисленной переменной size	2
2		Ввод значения переменной size	3
3	size <= 2 или значение переменной size нечётное	Вывод значения переменной size со знаком вопроса	∅
			4
4		Вывод значения переменной size	5
5		Вывод переноса на новую строку	6
6		Создание объекта obj класса Class с параметром size	7
7		Вызов метода Input() объекта obj	8
8		Вызов функции Call(obj)	9
9		Вывод переноса на новую строку	10
10		Вызов метода SumPara() объекта obj	11
11		Вывод значения, возвратимого методом Sum() объекта obj	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

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

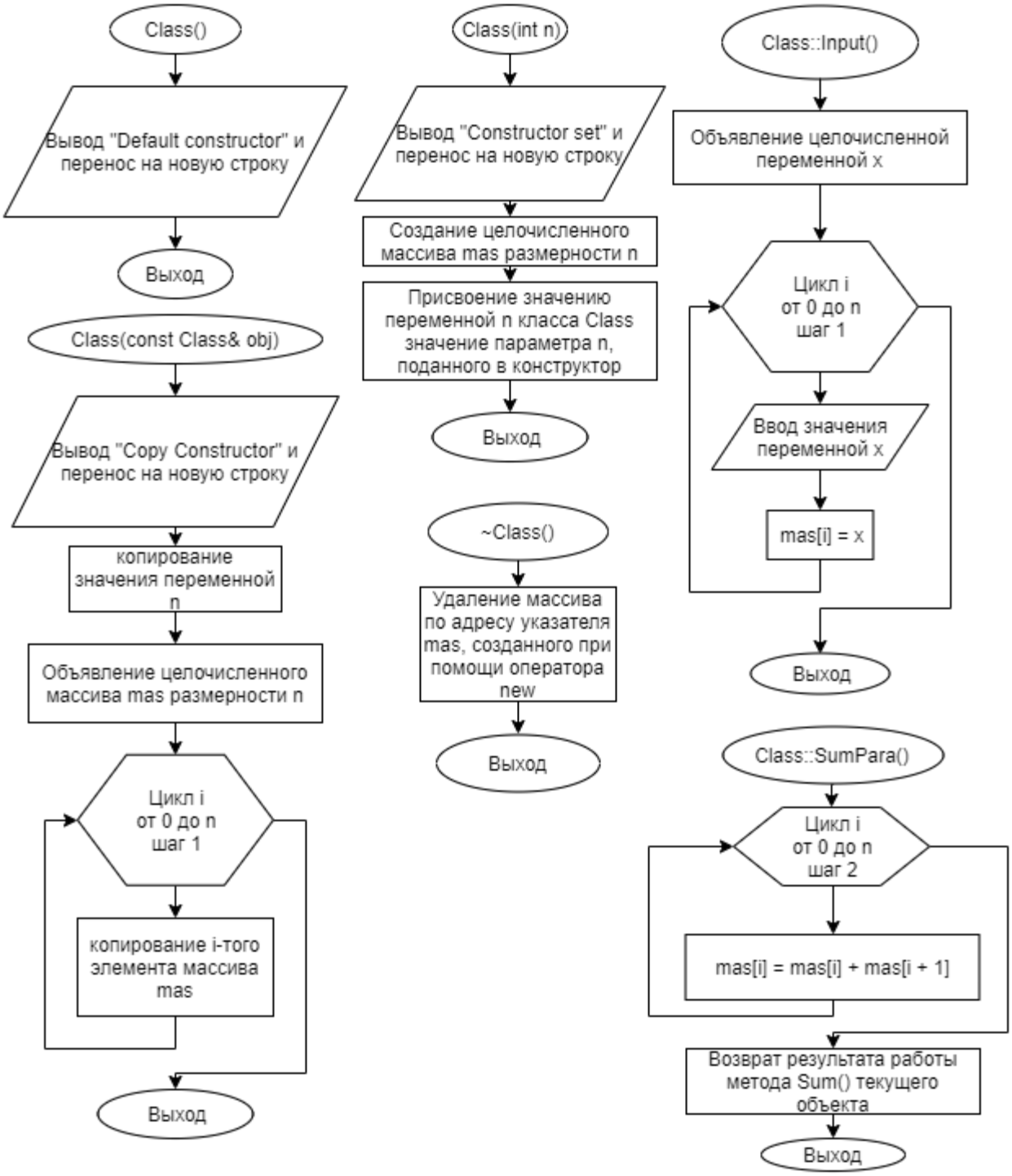

Рисунок 1 – Блок-схема алгоритма

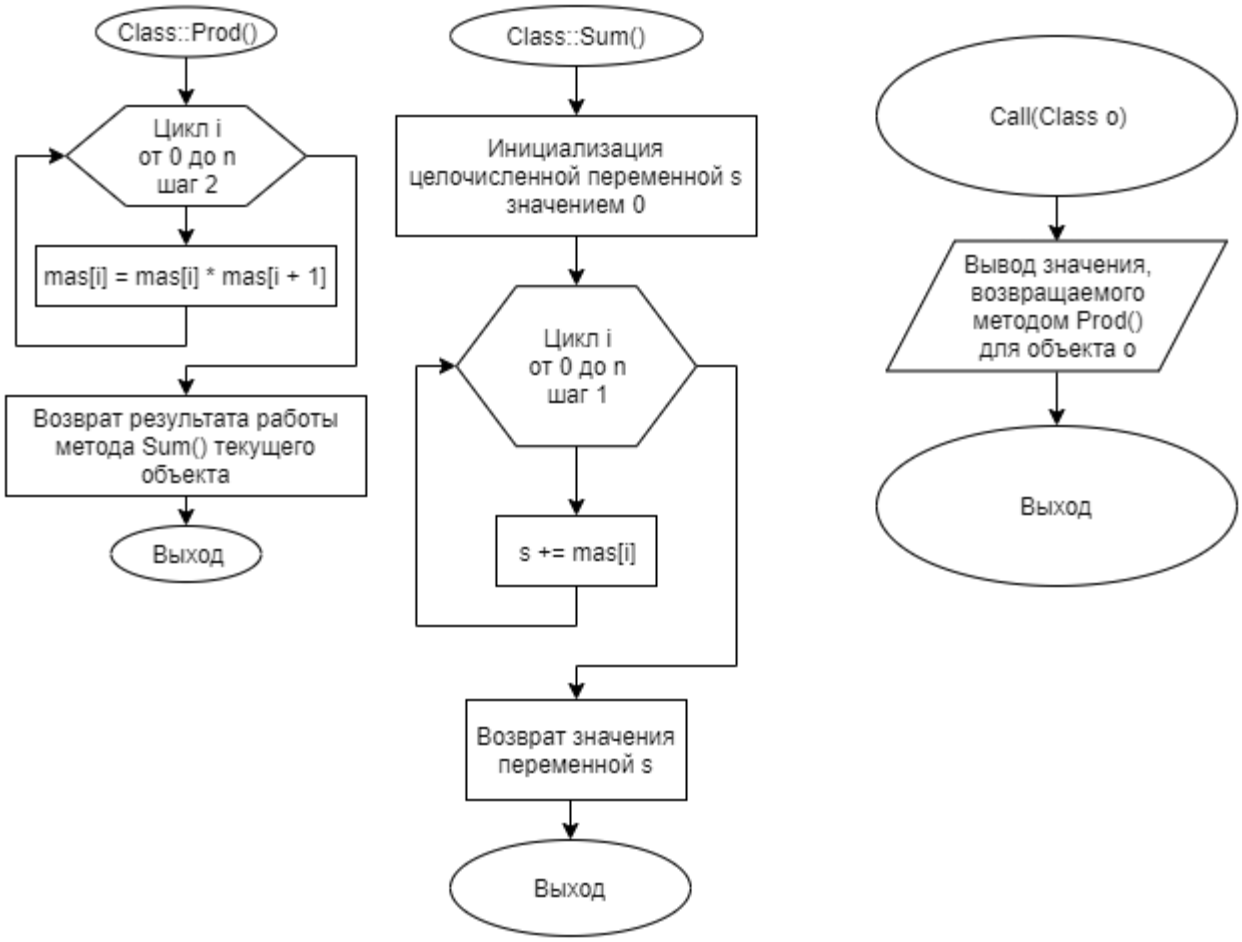

Рисунок 2 – Блок-схема алгоритма

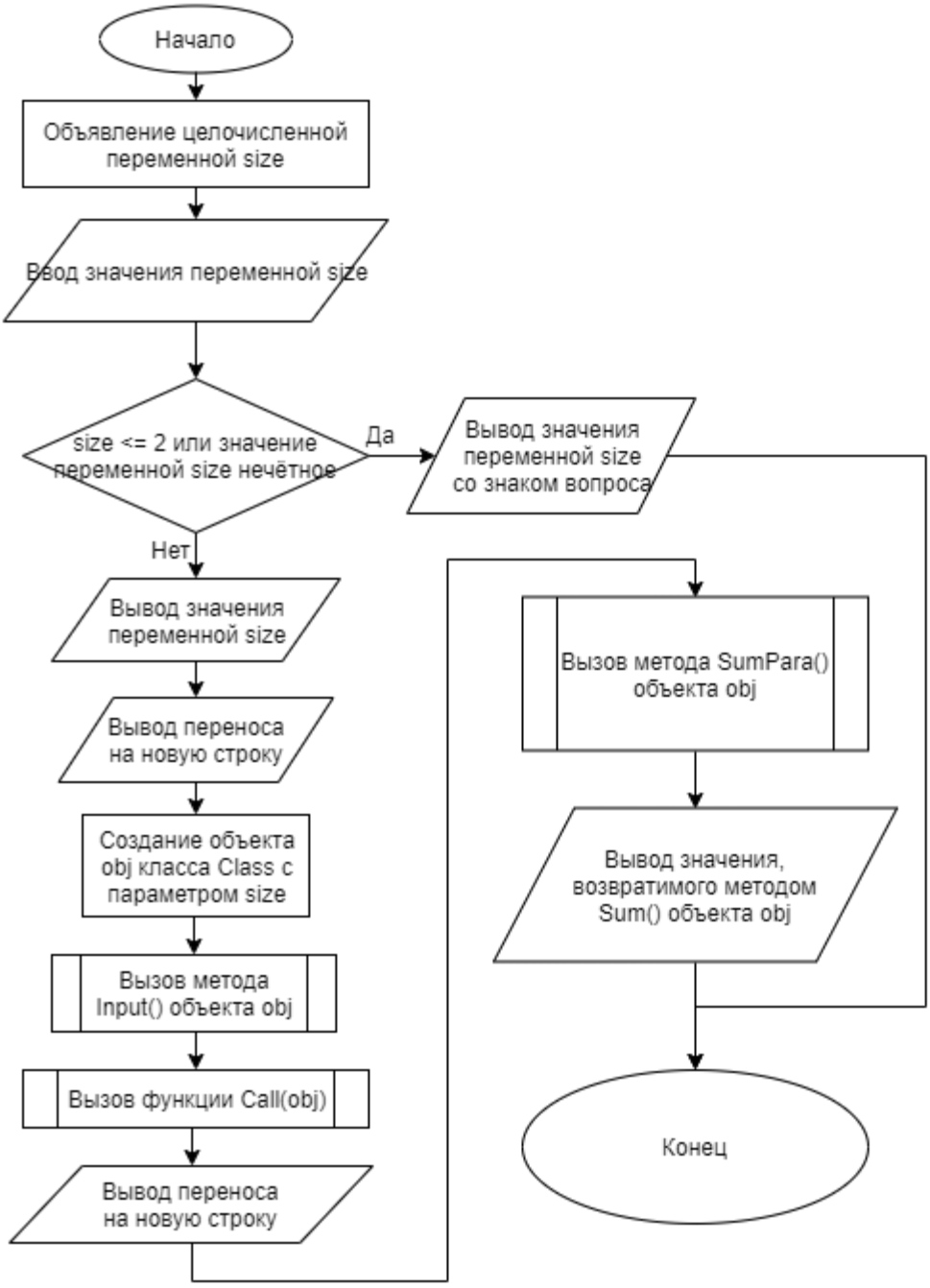

Рисунок 3 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл Class.cpp

Листинг 1 – Class.cpp

```
#include <iostream>
#include "Class.h"
using namespace std;

Class::Class() {
	cout << "Default constructor" << endl;
	mas = nullptr;
}
Class::Class(int n) {
	cout << "Constructor set" << endl;
	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;
	}
}
int Class::SumPara() {
	for (int i = 0; i < n; i += 2) {
		mas[i] = mas[i] + mas[i + 1];
	}
	return Sum();
}
int Class::Prod() {
```

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

5.2 Файл Class.h

Листинг 2 – Class.h

```
#ifndef __CLASS_H
#define __CLASS_H
class Class {
private:
	int n;
	int *mas;
public:
	Class();
	Class(int n);
	Class(const Class& obj);
	~Class();
	void Input();
	int SumPara();
	int Prod();
	int Sum();
};
#endif
```

5.3 Файл main.cpp

Листинг 3 – main.cpp

```
#include <iostream>
#include "Class.h"
using namespace std;

void Call(Class o) {
	cout << o.Prod();
}

int main() {
	int size;
```

```
	cin >> size;
	if (size <= 2 || size % 2 != 0) {
		cout << size << "?";
		return 0;
	}
	cout << size;
	cout << endl;
	Class obj(size);
	obj.Input();
	Call(obj);
	cout << endl;
	obj.SumPara();
	cout << obj.Sum();
}
```

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
5 1 2 3 4 5	5?	5?
2 1 2	2?	2?

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Васильев А.Н. Объектно-ориентированное программирование на C++. Издательство: Наука и Техника. Санкт-Петербург, 2016г. 543 стр.

2. Шилдт Г. C++: базовый курс. 3-е изд. Пер. с англ.. — М.: Вильямс, 2017. — 624 с.

3. Методическое пособие для проведения практических заданий, контрольных и курсовых работ по дисциплине «Объектно-ориентированное программирование» [Электронный ресурс] – URL: https://mirea.aco-avrora.ru/student/files/methodichescoe_posobie_dlya_laboratorny h_rabot_3.pdf (дата обращения 05.05.2021).

4. 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).

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).