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:

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 obj класса obj предназначен для ;
- функция main для Основная функция;
- Цикл со счётчиком;
- Условный оператор;
- Объекты стандартного потока ввода/вывода cin/cout.

Класс obj:

- свойства/поля:
    - поле Массив:
        - наименование — arr;
        - тип — Указаетль на целочисленный;
        - модификатор доступа — private;
    - поле Размерность массива:
        - наименование — size_arr;
        - тип — Целочисленный;
        - модификатор доступа — private;
- функционал:
    - метод obj — Конструктор;
    - метод obj_p — Параметризированный конструктор, по значению которого определяется размерность массива;
    - метод obj_c — Конструктор копирования;
    - метод ~obj — Деструктор, выводящий на экран сообщение и очищающий память, выделенную под массива;
    - метод Fill — Используется для заполнения массива;
    - метод Sum — Предназначен для суммирования значений элементов

массива;

- метод Metod1 — Используется для суммирования очередной пары элементов массива и присваивания значения первому элементу массива;
- метод Metod2 — Используется для перемножения очередной пары элементов массива и присваивания значения первому элементу 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 obj

Функционал: Конструктор.

Параметры: нет.

Алгоритм конструктора представлен в таблице 1.

*Таблица 1 – Алгоритм конструктора класса obj*

| № | Предикат | Действия | № перехода |
|---|---|---|---|
| 1 | | Вывод "Default constructor" | ∅ |

## 3.2 Алгоритм деструктора класса obj

Функционал: Деструктор, выводящий на экран сообщение и очищающий память, выделенную под массива.

Параметры: нет.

Алгоритм деструктора представлен в таблице 2.

*Таблица 2 – Алгоритм деструктора класса obj*

| № | Предикат | Действия | № перехода |
|---|---|---|---|
| 1 | | Вывод "Destructor" | 2 |
| 2 | | Очищенние выделенной памяти под массива с помощью delete | ∅ |

## 3.3 Алгоритм метода Fill класса obj

Функционал: Используется для заполнения массива.

Параметры: нет.

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 3.

*Таблица 3 – Алгоритм метода Fill класса obj*

| № | Предикат | Действия | № перехода |
|---|---|---|---|
| 1 | | Инициализация целочисленной переменной i | 2 |
| 2 | i < size_arr | Ввод элемента массива | 3 |
| | | | ∅ |
| 3 | | Инкрементируем i | 2 |

## 3.4 Алгоритм метода Sum класса obj

Функционал: Предназначен для суммирования значений элементов массива.

Параметры: нет.

Возвращаемое значение: Целочисленное.

Алгоритм метода представлен в таблице 4.

*Таблица 4 – Алгоритм метода Sum класса obj*

| № | Предикат | Действия | № перехода |
|---|---|---|---|
| 1 | | Инициализация целочисленных переменных sum = 0, i = 0 | 2 |
| 2 | i < size_arr | Прибавляем к sum значение элемента массива | 3 |
| | | Возвращаем значение переменной sum | ∅ |
| 3 | | Инкрементируем i | 2 |

## 3.5 Алгоритм метода Metod1 класса obj

Функционал: Используется для суммирования очередной пары элементов массива и присваивания значения первому элементу массива.

Параметры: нет.

Возвращаемое значение: Целочисленное.

Алгоритм метода представлен в таблице 5.

*Таблица 5 – Алгоритм метода Metod1 класса obj*

| № | Предикат | Действия | № перехода |
|---|---|---|---|
| 1 | | Инициализация целочисленной переменной i = 0 | 2 |
| 2 | i < size_arr | Суммируем пару элементов массива и присваиваем это значение первому элементу пары | 3 |
| | | Возвращаем метод Sum() | ∅ |
| 3 | | i++ | 2 |

## 3.6 Алгоритм метода Metod2 класса obj

Функционал: Используется для перемножения очередной пары элементов массива и присваивания значения первому элементу массива.

Параметры: нет.

Возвращаемое значение: Целочисленное.

Алгоритм метода представлен в таблице 6.

*Таблица 6 – Алгоритм метода Metod2 класса obj*

| № | Предикат | Действия | № перехода |
|---|---|---|---|
| 1 | | Инициализация целочисленной переменной i = 0 | 2 |
| 2 | i < size_arr | Перемножаем пару элементов массива и присваиваем это значение первому элементу пары | 3 |
| | | Возвращем метод Sum() | ∅ |

| № | Предикат | Действия | № перехода |
|---|---|---|---|
| 3 | | Инкрементируем i | 2 |

## 3.7 Алгоритм функции main

Функционал: Основная функция.

Параметры: Нет.

Возвращаемое значение: Целочисленное.

Алгоритм функции представлен в таблице 7.

*Таблица 7 – Алгоритм функции main*

| № | Предикат | Действия | № перехода |
|---|---|---|---|
| 1 | | Объявление целочисленной переменной size_arr | 2 |
| 2 | | Ввод значения size_arr | 3 |
| 3 | size_arr > 2 или size_arr четный | Вывод значения переменной size_arr | 4 |
| | | Вывод значения переменной size_arr и "?" | ∅ |
| 4 | | Создание объекта obj класс obj путем вызова параметризированного конструктора с аргументом size_arr | 5 |
| 5 | | Вызов метода Fill() | 6 |
| 6 | | Вызов функции fun(obj) | 7 |
| 7 | | Переход на новую строку | 8 |
| 8 | | Вызов метода Metod1 объекта obj | 9 |
| 9 | | Вывод результата выполнения метода Metod1 объекта obj | ∅ |

## 3.8 Алгоритм функции fun

Функционал: Вызов и вывод Метод2 объекта obj.

Параметры: obj.

Возвращаемое значение: Отсутствует.

Алгоритм функции представлен в таблице 8.

*Таблица 8 – Алгоритм функции fun*

| № | Предикат | Действия | № перехода |
|---|---|---|---|
| 1 | | Вызов Metod2 | 2 |
| 2 | | Вывод Metod2 | ∅ |

## 3.9 Алгоритм метода obj_p класса obj

Функционал: Параметризированный конструктор, по значению которого определяется размерность массива.

Параметры: нет.

Возвращаемое значение: Отсутствует.

Алгоритм метода представлен в таблице 9.

*Таблица 9 – Алгоритм метода obj_p класса obj*

| № | Предикат | Действия | № перехода |
|---|---|---|---|
| 1 | | Параметизированный конструктор с size_rr | ∅ |

## 3.10 Алгоритм конструктора класса obj

Функционал: Конструктор копирования.

Параметры: нет.

Алгоритм конструктора представлен в таблице 10.

*Таблица 10 – Алгоритм конструктора класса obj*

| № | Предикат | Действия | № перехода |
|---|---|---|---|
| 1 | | Конструктор копирования | ∅ |

# 4 БЛОК-СХЕМЫ АЛГОРИТМОВ

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

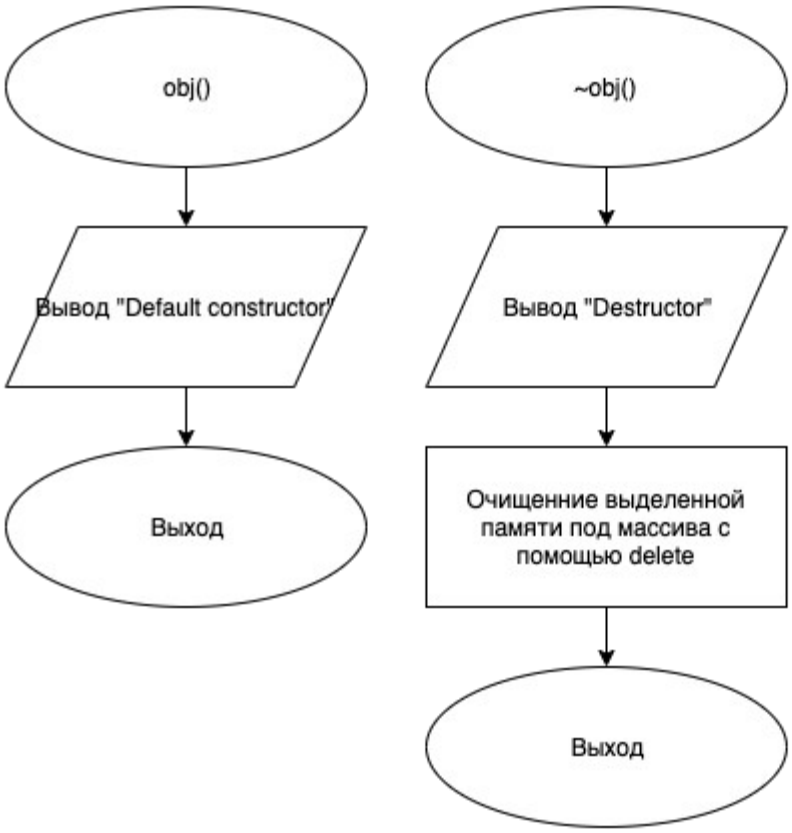


**Рисунок 1 – Блок-схема алгоритма**

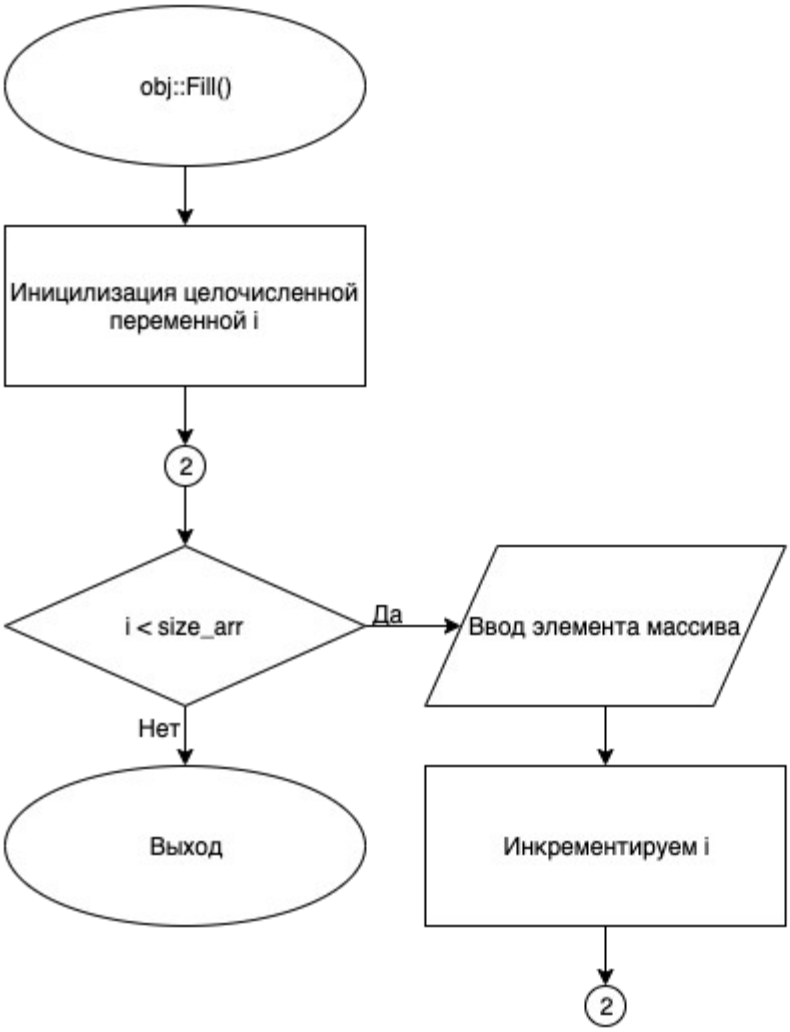


**Рисунок 2 – Блок-схема алгоритма**

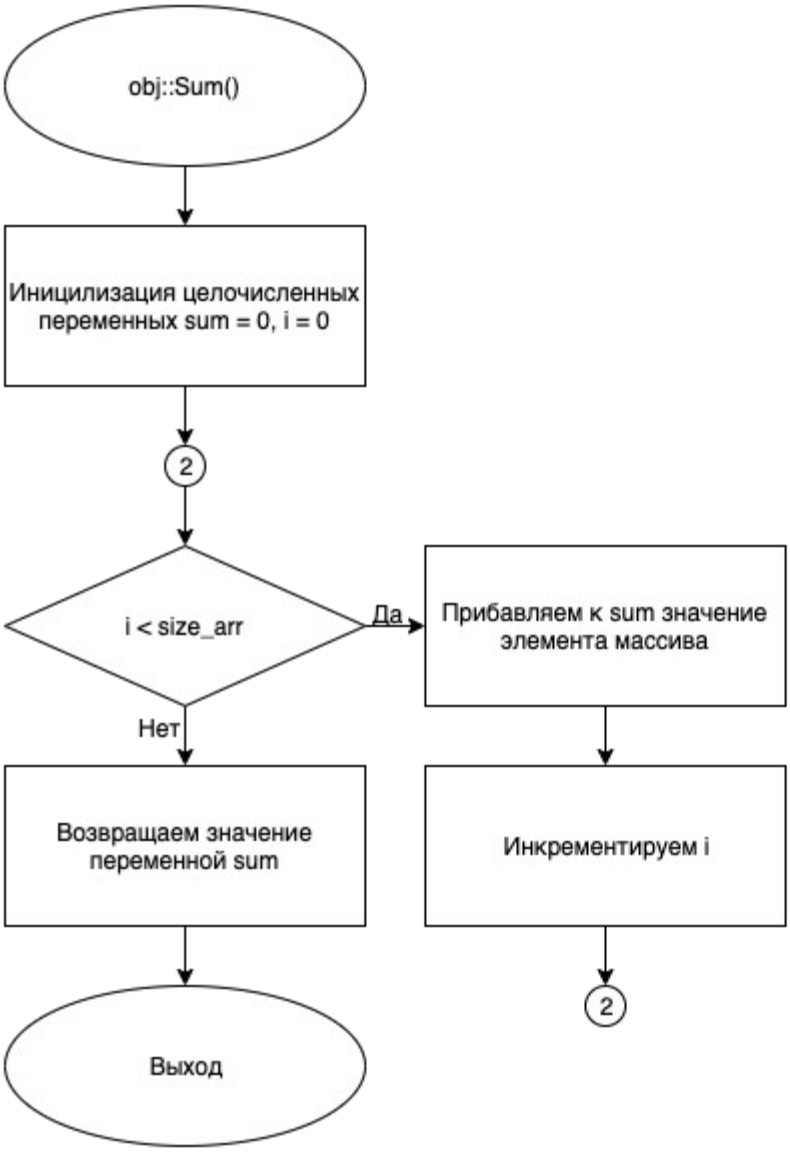


**Рисунок 3 – Блок-схема алгоритма**

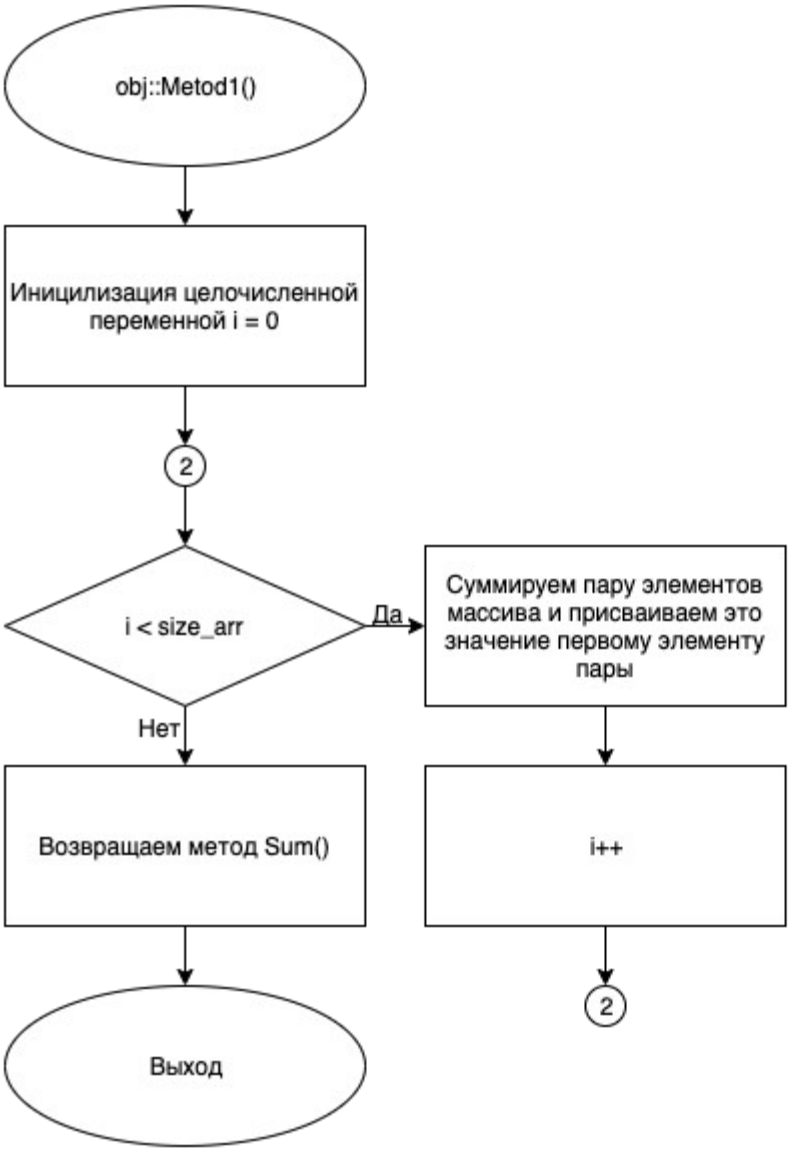


**Рисунок 4 – Блок-схема алгоритма**

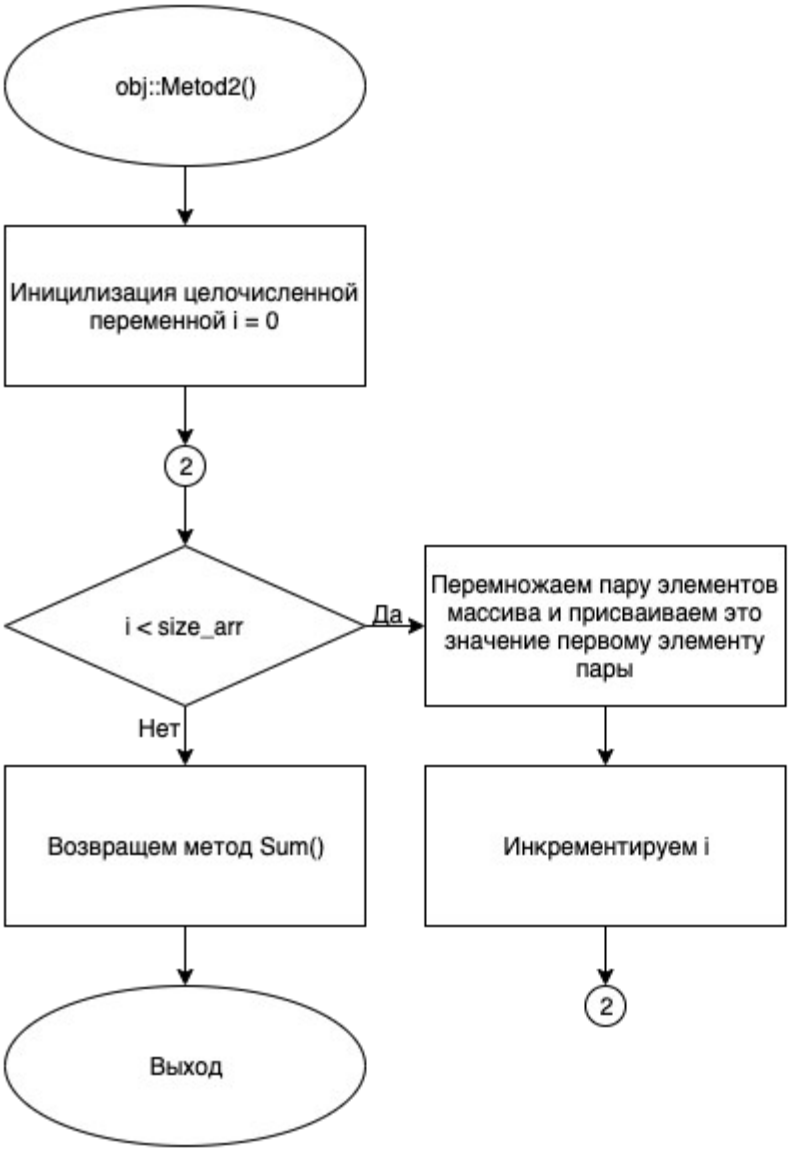


**Рисунок 5 – Блок-схема алгоритма**

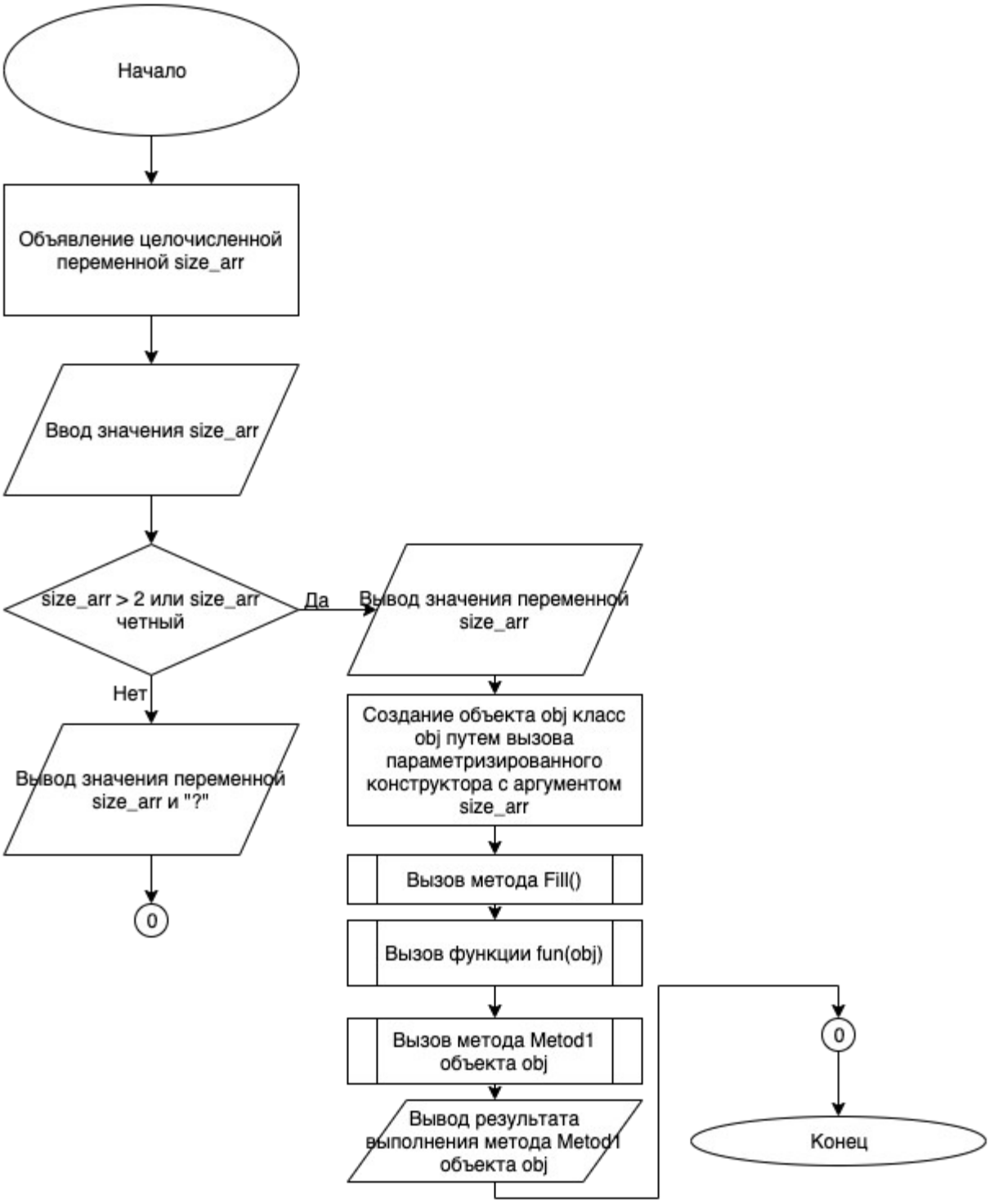


**Рисунок 6 – Блок-схема алгоритма**

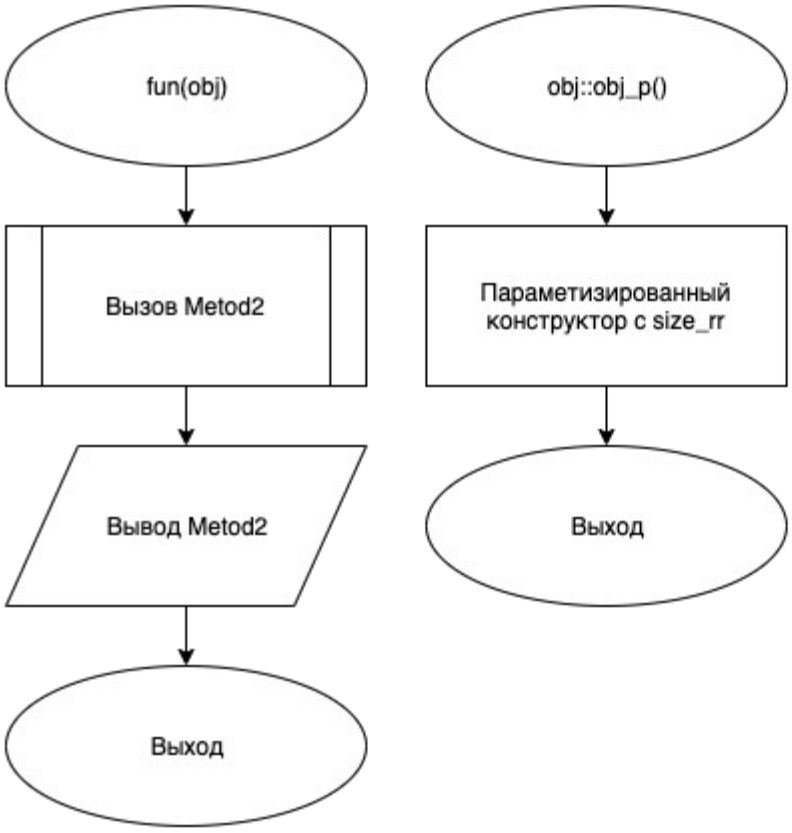


**Рисунок 7 – Блок-схема алгоритма**


**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 "obj.h"

using namespace std;
void fun(obj obj)
{
   cout << obj.Metod2() << endl;
}
int main()
{
   int size_arr;
   cin >> size_arr;
   if(size_arr > 2 && size_arr %2 == 0)
   {
      cout << size_arr << endl;
      obj obj(size_arr);
      obj.Fill();
      fun(obj);
      cout << endl;
      cout << obj.Metod1() << endl;
   }
   else
   {
      cout << size_arr << "?";
   }
   return(0);
}
```

## 5.2 Файл obj.cpp

*Листинг 2 – obj.cpp*

```
#include "obj.h"
```

```
#include <iostream>
using namespace std;
obj::obj()
{
   cout << "Default constructor" << endl;
}

obj::obj(int size_arr)
{
   cout << "Constructor set" << endl;
   arr = new int[size_arr];
   this->size_arr = size_arr;
}
obj::obj(const obj & obj)
{
   cout << "Copy constructor" << endl;
   size_arr = obj.size_arr;
   arr = new int[size_arr];
   for(int i = 0; i < size_arr; i ++)
   {
      arr[i] = obj.arr[i];
   }
}
obj::~obj()
{
   cout << "Destructor";
   if(arr == nullptr)
   {
      delete[] arr;
   }
}
void obj::Fill()
{
   for(int i = 0; i <size_arr; i++)
   {
      cin >> arr[i];
   }
}

int obj::Sum()
{
   int sum = 0;
   for(int i = 0; i < size_arr; i++)
   {
      sum += arr[i];
   }
   return sum;
}

int obj::Metod1()
{
   for(int i = 0; i < size_arr; i+=2)
   {
      arr[i] += arr[i+1];
```

```
   }
   return Sum();
}

int obj::Metod2()
{
   for(int i = 0; i < size_arr; i+=2)
   {
      arr[i] *= arr[i+1];
   }
   return Sum();
}
```

## 5.3 Файл obj.h

*Листинг 3 – obj.h*

```
#ifndef __IKBO51__H
#define __IKBO51__H

class obj
{
private:
   int* arr;
   int size_arr;
public:
   obj();
   obj(int size_arr);
   obj(const obj & obj);
   ~obj();
   void Fill();
   int Sum();
   int Metod1();
   int Metod2();
};

#endif
```

# 6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 11.

*Таблица 11 – Результат тестирования программы*

| **Входные данные** | **Ожидаемые выходные данные** | **Фактические выходные данные** |
|---|---|---|
| 2 | 2? | 2? |
| 4<br>1 2 3 4 | 4<br>Constructor set<br>Copy constructor<br>20<br>Destructor<br>16<br>Destructor | 4<br>Constructor set<br>Copy constructor<br>20<br>Destructor<br>16<br>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-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).