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 object класса obj предназначен для основной работы с методами класса;
- объект ob_local класса obj предназначен для работы с копией объекта;
- функция main для основная функция программы;
- функция func для функция для создания копии объекта и работы с ней;
- cin - объект стандартного потока ввода с клавиатуры;
- cout - объект стандартного потока вывода на экран;
- if..else - условный оператор;
- for - оператор цикла со счётчиком;
- new - динамическое выделение памяти;
- delete - динамическое очищение памяти.

Класс obj:

- свойства/поля:
    - поле размерность массива объекта:
        - наименование — n;
        - тип — int;
        - модификатор доступа — private;
    - поле массив объекта:
        - наименование — matrix;
        - тип — int*;
        - модификатор доступа — private;
- функционал:
    - метод obj — конструктор по умолчанию;
    - метод obj — параметризированный конструктор;

- метод obj — конструктор копии объекта;
- метод ~obj — деструктор;
- метод input — ввод значений в массив объекта;
- метод sum — вычисляет сумму пар элементов объекта, записывает результат в первый элемент пары, повторяет эти действия со следующими парами, затем вычисляет сумму всего массива и возвращает его значение;
- метод mult — вычисляет произведение пар элементов объекта, записывает результат в первый элемент пары, повторяет эти действия со следующими парами, затем вычисляет произведение всего массива и возвращает его значение;
- метод output — вычисляет сумму всех элементов массива и возвращает результат.

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 по умолчанию.

Параметры: нет.

Алгоритм конструктора представлен в таблице 1.

Таблица 1 – Алгоритм конструктора класса obj

№	Предикат	Действия	№ перехода
1		вывод на экран сообщения: "\nDefault constructor"	∅

3.2 Алгоритм деструктора класса obj

Функционал: деструктор.

Параметры: нет.

Алгоритм деструктора представлен в таблице 2.

Таблица 2 – Алгоритм деструктора класса obj

№	Предикат	Действия	№ перехода
1		вывод на экран сообщения: "\nDestructor"	2
2		динамическое очищение памяти от массива объекта	∅

3.3 Алгоритм метода input класса obj

Функционал: ввод значений в массив объекта.

Параметры: нет.

Возвращаемое значение: void.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода input класса obj

№	Предикат	Действия	№ перехода
1		инициализация переменной i типа int значением 0	2
2	i < n	ввод значения элемента массива объекта с индексом i	3
			∅
3		увеличение значения i на 1	2

3.4 Алгоритм метода sum класса obj

Функционал: вычисляет сумму пар элементов объекта, записывает результат в первый элемент пары, повторяет эти действия со следующими парами, затем вычисляет сумму всего массива и возвращает его значение.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода sum класса obj

№	Предикат	Действия	№ перехода
1		инициализация переменной i типа int значением 0	2
2	i < n-1	увеличение значения элемента массива объекта с индексом i на значение элемента массива объекта с индексом i+1	3
		инициализация переменной sum типа int значением 0	4
3		увеличение значения переменной i на 2	2

№	Предикат	Действия	№ перехода
4		присваивание переменной i значение 0	5
5	i < n	увеличение значения переменной sum на значение элемента массива объекта с индексом i	6
		возврат значения sum	∅
6		увеличение значения переменной i на 1	5

3.5 Алгоритм метода mult класса obj

Функционал: вычисляет произведение пар элементов объекта, записывает результат в первый элемент пары, повторяет эти действия со следующими парами, затем вычисляет произведение всего массива и возвращает его значение.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода mult класса obj

№	Предикат	Действия	№ перехода
1		инициализация переменной i типа int значением 0	2
2	i < n-1	умножение значения элемента массива объекта с индексом i на значение элемента массива объекта с индексом i+1	3
		инициализация переменной mult типа int значением 1	4
3		увеличение значения переменной i на 2	2
4		присваивание переменной i значение 0	5
5	i < n	умножение значения переменной mult на значение элемента массива объекта с индексом i	6
		возврат значения mult	∅
6		увеличение значения переменной i на 1	5

3.6 Алгоритм метода output класса obj

Функционал: вычисляет сумму всех элементов массива и возвращает результат.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 6.

Таблица 6 – Алгоритм метода output класса obj

№	Предикат	Действия	№ перехода
1		инициализация переменной sum типа int значением 0	2
2		инициализация переменной i типа int значением 0	3
3	i < n	увеличение значения переменной sum на значение элемента массива объекта с индексом i	4
		возврат значение переменной sum	∅
4		увеличение значение переменной i на 1	3

3.7 Алгоритм конструктора класса obj

Функционал: параметризированный конструктор.

Параметры: int n.

Алгоритм конструктора представлен в таблице 7.

Таблица 7 – Алгоритм конструктора класса obj

№	Предикат	Действия	№ перехода
1		вывод на экран сообщения: "\nConstructor set"	2
2		запись в поле размерности объекта значение полученного параметра	3
3		создание динамической матрицы размерности параметра	∅

3.8 Алгоритм конструктора класса obj

Функционал: конструктор копии.

Параметры: const obj & ob.

Алгоритм конструктора представлен в таблице 8.

Таблица 8 – Алгоритм конструктора класса obj

№	Предикат	Действия	№ перехода
1		вывод на экран сообщения: "\nCopy constructor"	2
2		присваивание полю размерности массива копии объекта значение поля размерности массива передаваемого объекта	3
3		динамическое выделение памяти под массив копии передаваемого объекта	4
4		инициализация переменной i типа int значением 0	5
5	i < n	присваивание элементу массива копии объекта с индексом i значение элемента массива передаваемого объекта с индексом i	6
			∅
6		увеличение значения переменной i на 1	5

3.9 Алгоритм функции main

Функционал: основная функция программы.

Параметры: нет.

Возвращаемое значение: int - код ошибки.

Алгоритм функции представлен в таблице 9.

Таблица 9 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		объявление переменной n типа int	2
2		ввод значения переменной n	3
3	n > 2 && n % 2 == 0	вывод на экран значение переменной n	4
		вывод на экран сообщения n "?"	10
4		объявление объекта object класса obj с помощью параметризированного конструктора с аргументом n	5
5		вызов метода input объекта object	6
6		создание копии объекта object с помощью функции func и конструктора копии объекта	7
7		вызов метода sum объекта object	8
8		вывод на экран перехода на новую строку	9
9		вызов метода output объекта object	10
10		возврат значения 0	∅

3.10 Алгоритм функции func

Функционал: работа с копией объекта.

Параметры: obj ob_local.

Возвращаемое значение: void.

Алгоритм функции представлен в таблице 10.

Таблица 10 – Алгоритм функции func

№	Предикат	Действия	№ перехода
1		вызов метода mult объекта ob_local	2
2		вывод на экран перехода на новую строку	3
3		вызов метода output объекта ob_local	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-6.

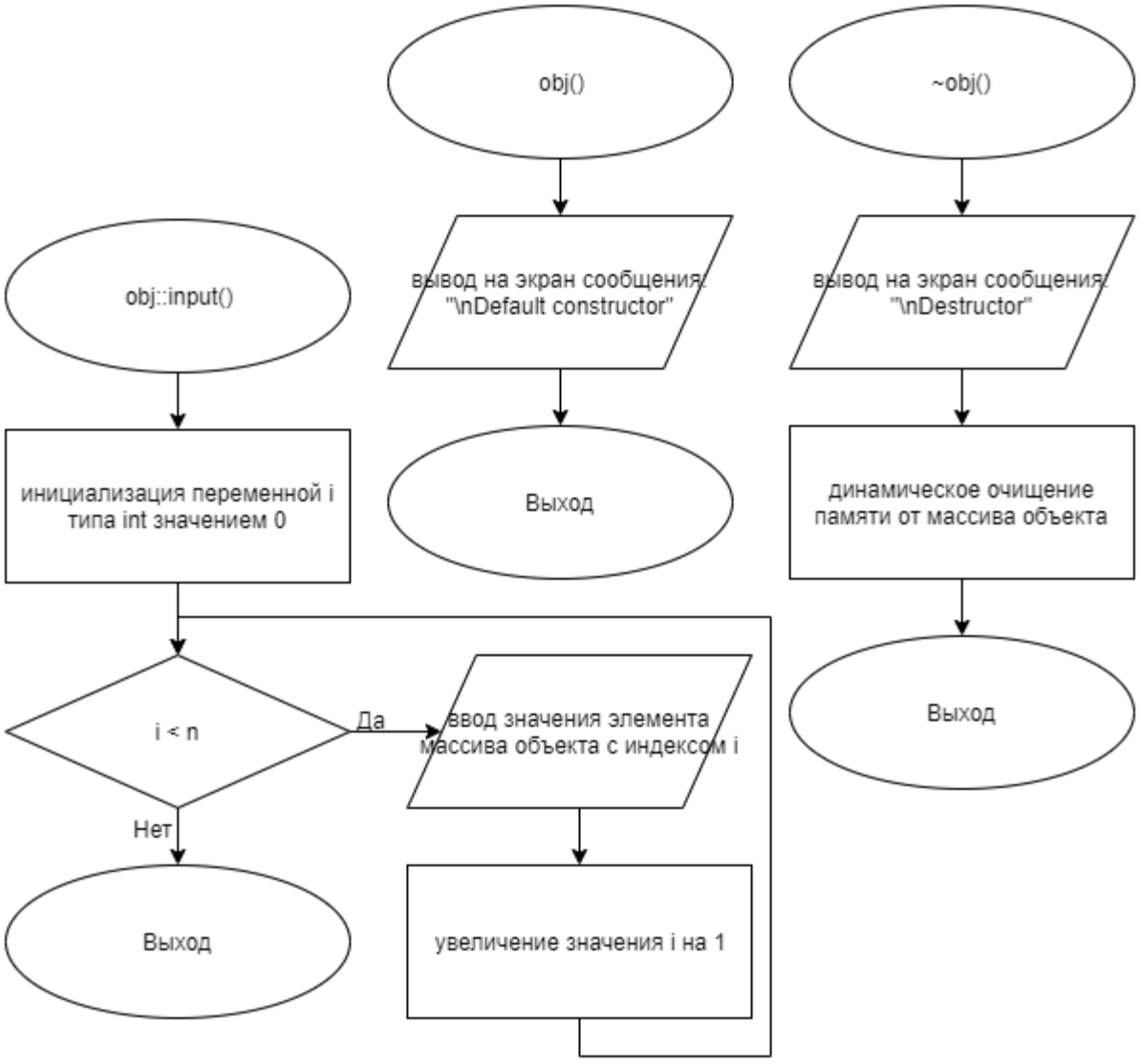

Рисунок 1 – Блок-схема алгоритма

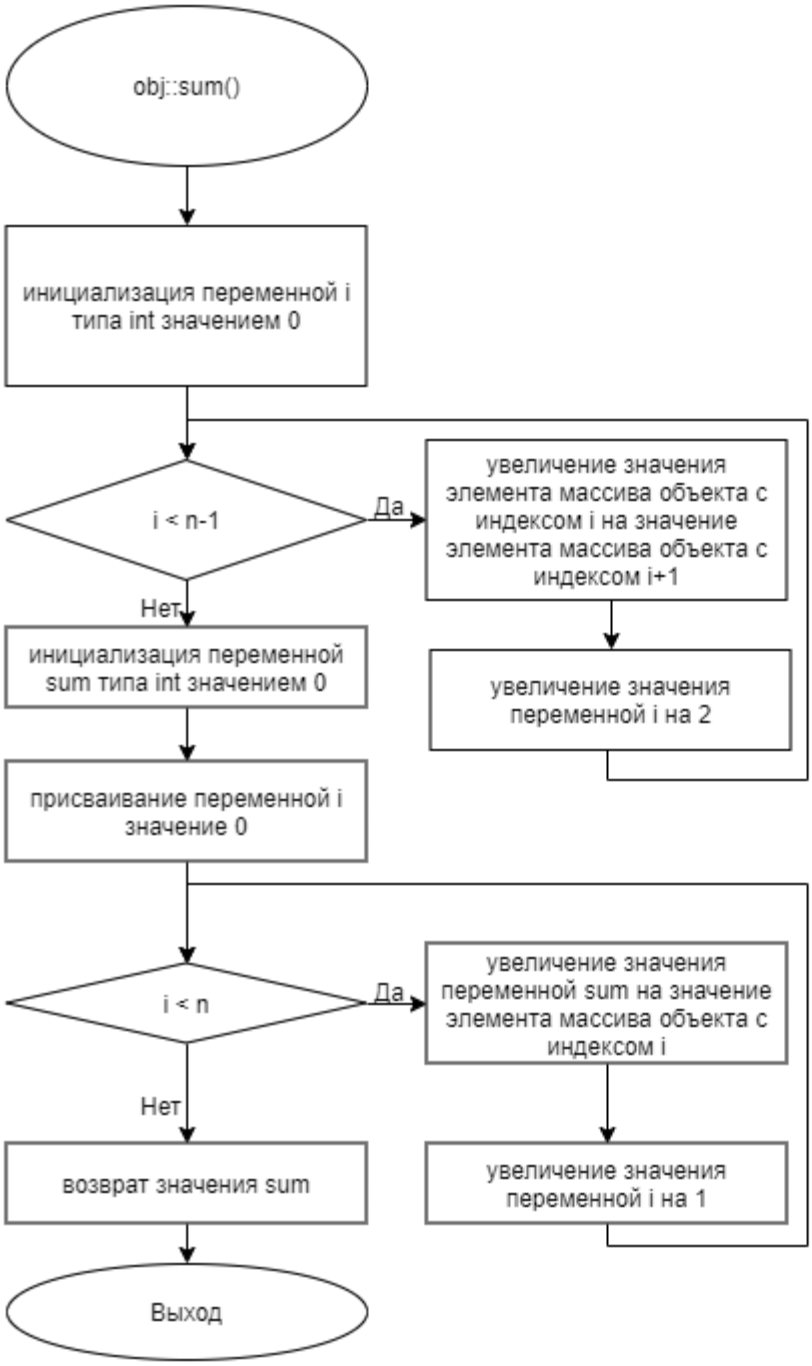

Рисунок 2 – Блок-схема алгоритма

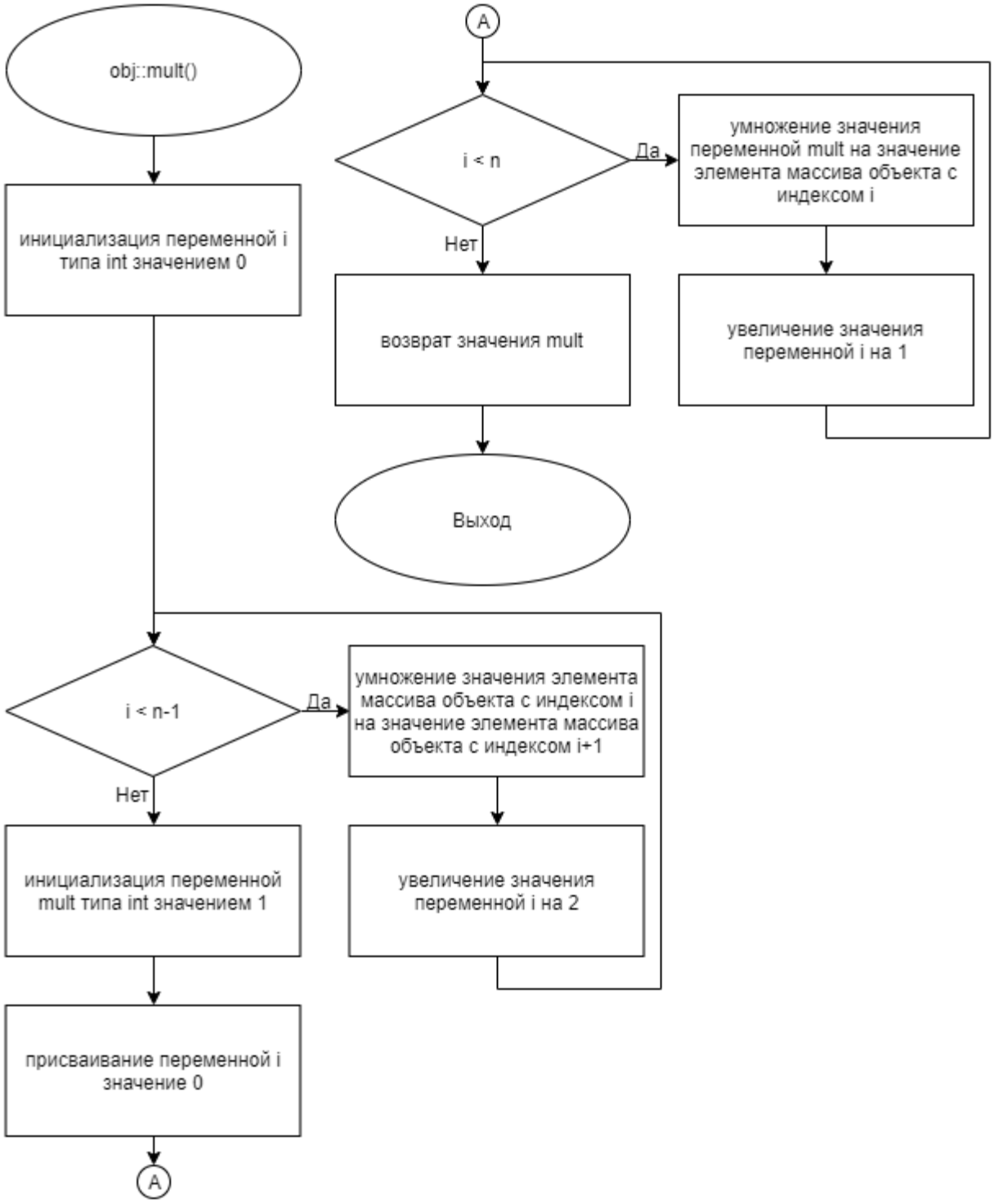

Рисунок 3 – Блок-схема алгоритма

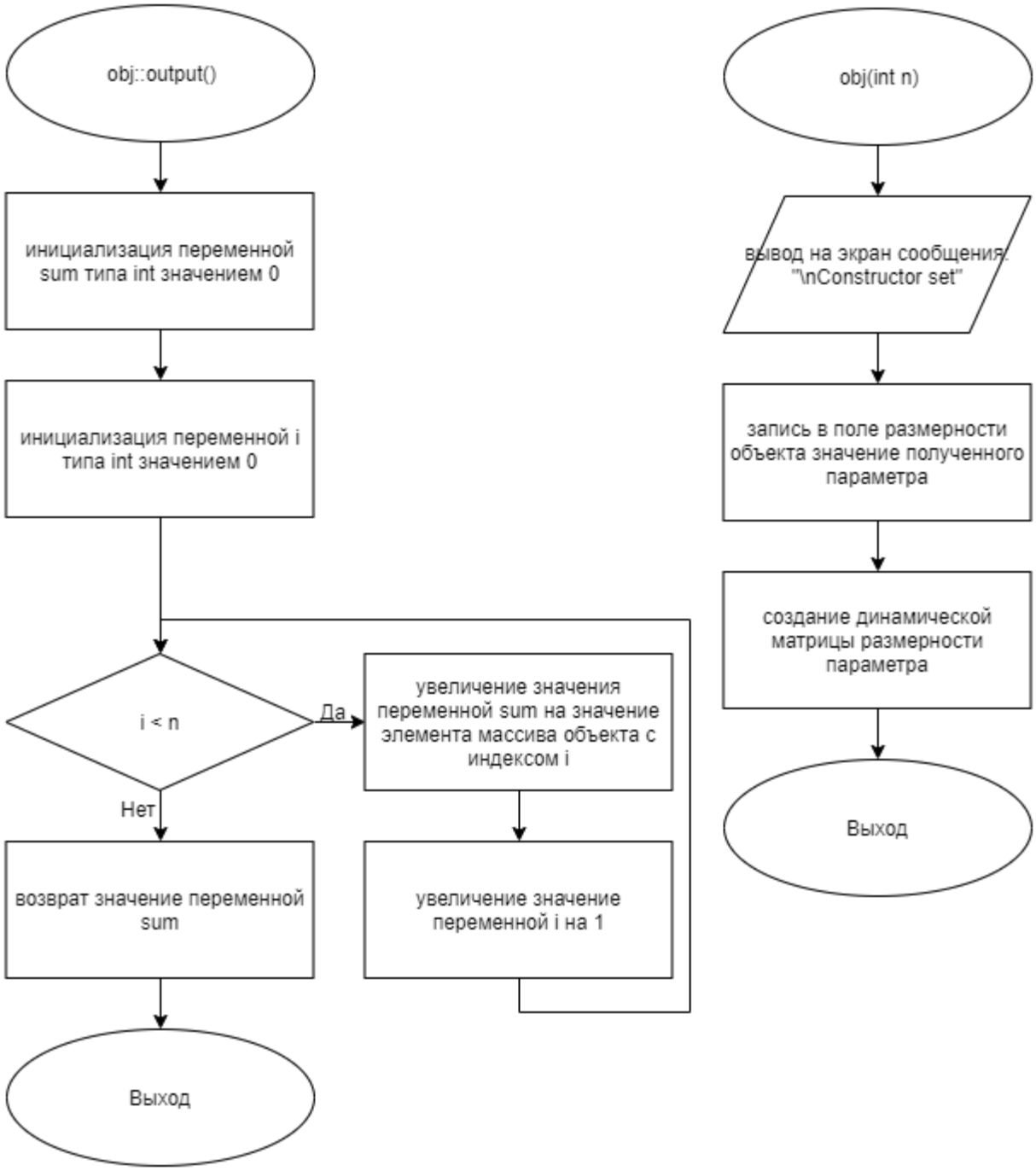

Рисунок 4 – Блок-схема алгоритма

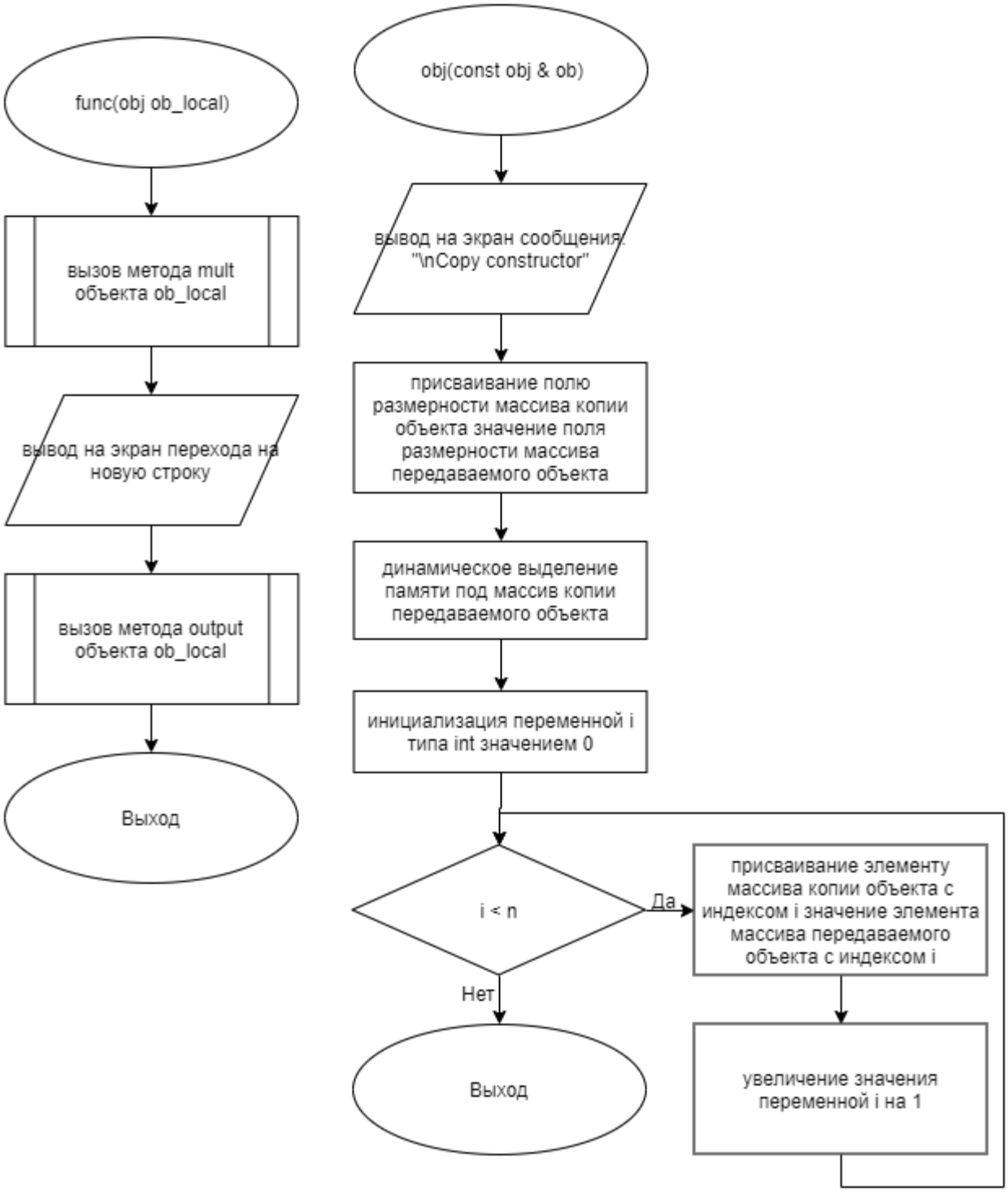

Рисунок 5 – Блок-схема алгоритма

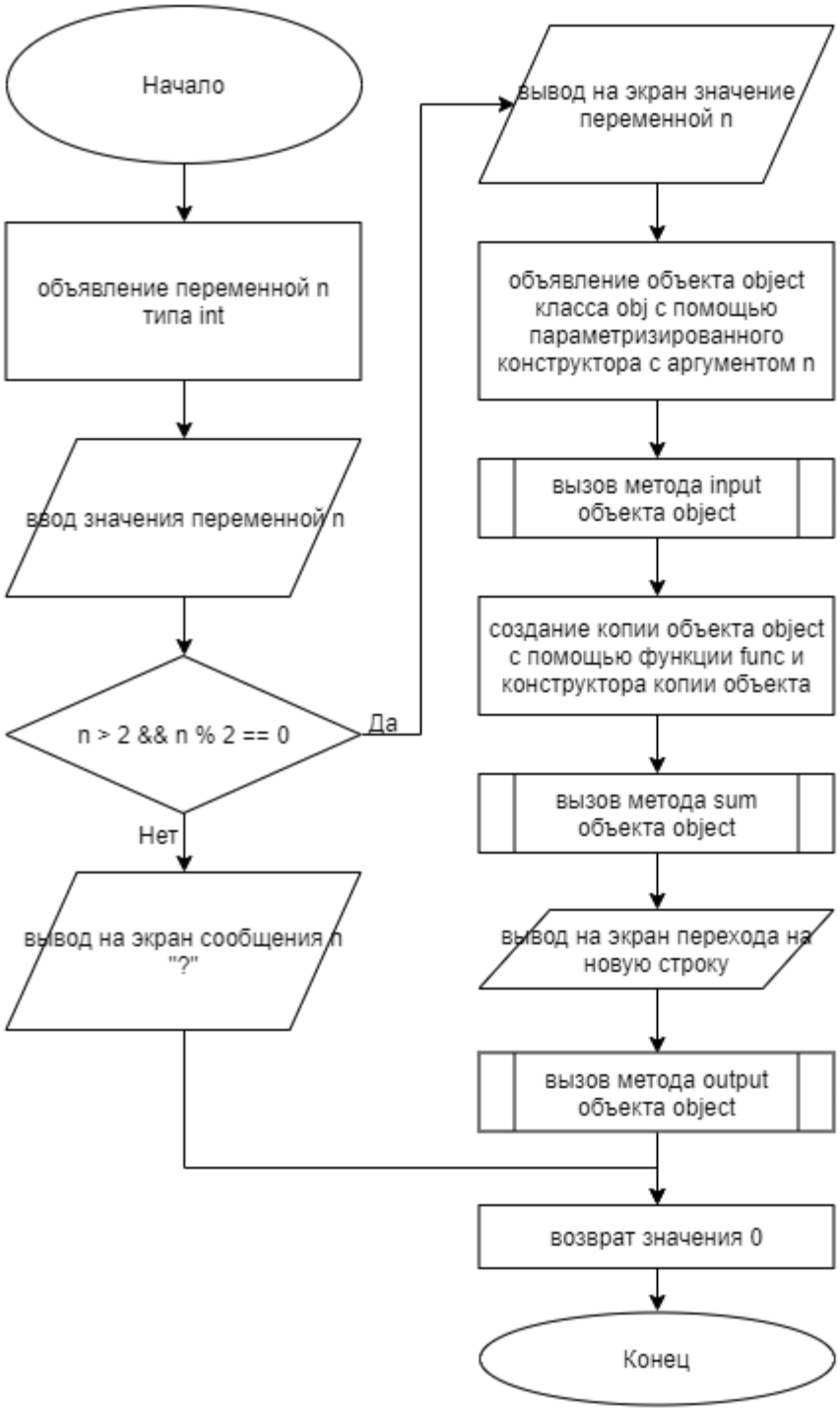

Рисунок 6 – Блок-схема алгоритма

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"

void func(obj ob_local)
{
   ob_local.mult();
   std::cout << std::endl << ob_local.output();
}

int main()
{
   int n;
   std::cin >> n;
   if (n > 2 && n % 2 == 0)
   {
      std::cout << n;
      obj object(n);
      object.input();
      func (object);
      object.sum();
      std::cout << std::endl << object.output();
   }
   else
      std::cout << n << '?';
   return(0);
}
```

5.2 Файл obj.cpp

Листинг 2 – obj.cpp

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

```
#include <iostream>

obj::obj()
{
   std::cout << "\nDefault constructor";
}
obj::obj(int n)
{
   std::cout << "\nConstructor set";
   this -> n = n;
   matrix = new int[n];
}
obj::obj(const obj & ob)
{
   std::cout << "\nCopy constructor";
   n = ob.n;
   matrix = new int [n];
   for (int i = 0; i < n; i++)
      matrix[i] = ob.matrix[i];
}
obj::~obj()
{
   std::cout << "\nDestructor";
   delete[] matrix;
}
void obj::input()
{
   for (int i = 0; i < n; i++)
      std::cin >> matrix[i];
}
int obj::sum()
{
   for (int i = 0; i < n-1; i+=2)
      matrix[i] += matrix[i+1];
   int sum = 0;
   for (int i = 0; i < n; i++)
      sum += matrix[i];
   return sum;
}
int obj::mult()
{
   for (int i = 0; i < n-1; i+=2)
      matrix[i] *= matrix[i+1];
   int mult = 1;
   for (int i = 0; i < n; i++)
      mult *= matrix[i];
   return mult;
}
int obj::output()
{
   int sum = 0;
   for (int i = 0; i < n; i++)
      sum += matrix[i];
   return sum;
}
```

5.3 Файл obj.h

Листинг 3 – obj.h

```
#ifndef __OBJ__H
#define __OBJ__H

class obj
{
private:
   int n;
   int *matrix = nullptr;
public:
   obj();
   obj(int n);
   obj(const obj & ob);
   ~obj();
   void input();
   int sum();
   int mult();
   int output();
};

#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?
7	7?	7?
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
4 10 23 62 71	4 Constructor set Copy constructor 4726 Destructor 260 Destructor	4 Constructor set Copy constructor 4726 Destructor 260 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).