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, вначале работы выдает сообщение;
- Параметризированный 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 параметризированного 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
```

Параметризированный 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 main для основная функция программы;
- функция func для создания локального объекта.

Класс obj:

- функционал:
    - метод create — метод, который создает целочисленный массив в закрытой области, согласно заранее созданной размерности;
    - метод print — метод последовательного вывода содержимого элементов массива, которые разделены тремя пробелами;
    - метод input — метод ввода данных;
    - метод sum — метод суммирования;
    - метод mult — метод умножения.

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 obj

Функционал: метод, который создает целочисленный массив в закрытой области, согласно заранее созданной размерности.

Параметры: нет.

Возвращаемое значение: void.

Алгоритм метода представлен в таблице 1.

Таблица 1 – Алгоритм метода create класса obj

№	Предикат	Действия	№ перехода
1		создание целочисленного массива в закрытой области, согласно заранее созданной размерности	∅

3.2 Алгоритм метода print класса obj

Функционал: метод последовательного вывода содержимого элементов массива, которые разделены тремя пробелами.

Параметры: нет.

Возвращаемое значение: void.

Алгоритм метода представлен в таблице 2.

Таблица 2 – Алгоритм метода print класса obj

№	Предикат	Действия	№ перехода
1		вывод на экран перхода на новую строку и первого элемента массива	2
2		иниицализация переменной i типа int значением 1	3
3	i < n	вывод на экран " " (три пробела) и значение элемента массива с индексом i	4
			∅
4		i++	3

3.3 Алгоритм функции main

Функционал: основная функция программы.

Параметры: нет.

Возвращаемое значение: int - код ошибки.

Алгоритм функции представлен в таблице 3.

Таблица 3 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		объявление переменной n типа int и ввод её значения с клавиатуры	2
2	n > 2 && n % 2 == 0	вывод на экран значение переменной n	3
		вывод на экран значение переменной n и '?'	14
3		создание объекта object_1 с помощью конструктора по умолчанию	4
4		присвоение object_1 результата работы функции func с аргументом n	5
5		вызов метода create() объекта object_1	6
6		вызов метода input() объекта object_1	7
7		вызов метода mult() объекта object_1	8

№	Предикат	Действия	№ перехода
8		инициализация объекта object_2 класса obj объектом object_1	9
9		вызов метода sum() объекта object_2	10
10		вызов метода print() объекта object_1	11
11		вызов метода output() объекта object_1	12
12		вызов метода print() объекта object_2	13
13		вызов метода output() объекта object_2	14
14		возврат 0	∅

3.4 Алгоритм функции func

Функционал: создаёт локальный объект.

Параметры: int n.

Возвращаемое значение: base_class.

Алгоритм функции представлен в таблице 4.

Таблица 4 – Алгоритм функции func

№	Предикат	Действия	№ перехода
1		создание объекта ob_local с помощью параметризированного конструктора объекта с параметром n	2
2		возврат ob_local	∅

3.5 Алгоритм метода input класса obj

Функционал: метод ввода данных.

Параметры: нет.

Возвращаемое значение: void.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода input класса obj

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

3.6 Алгоритм метода sum класса obj

Функционал: метод суммирования.

Параметры: нет.

Возвращаемое значение: void.

Алгоритм метода представлен в таблице 6.

Таблица 6 – Алгоритм метода sum класса obj

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

3.7 Алгоритм метода mult класса obj

Функционал: метод умножения.

Параметры: нет.

Возвращаемое значение: void.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода mult класса obj

№	Предикат	Действия	№ перехода
1		инициализация переменной i типа int значением 0	2
2	i < n-1	умножение значения элемента с индексом i на значение элемента с индексом i+1	3
			∅
3		увеличение значения перемееной i на 2	2

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

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

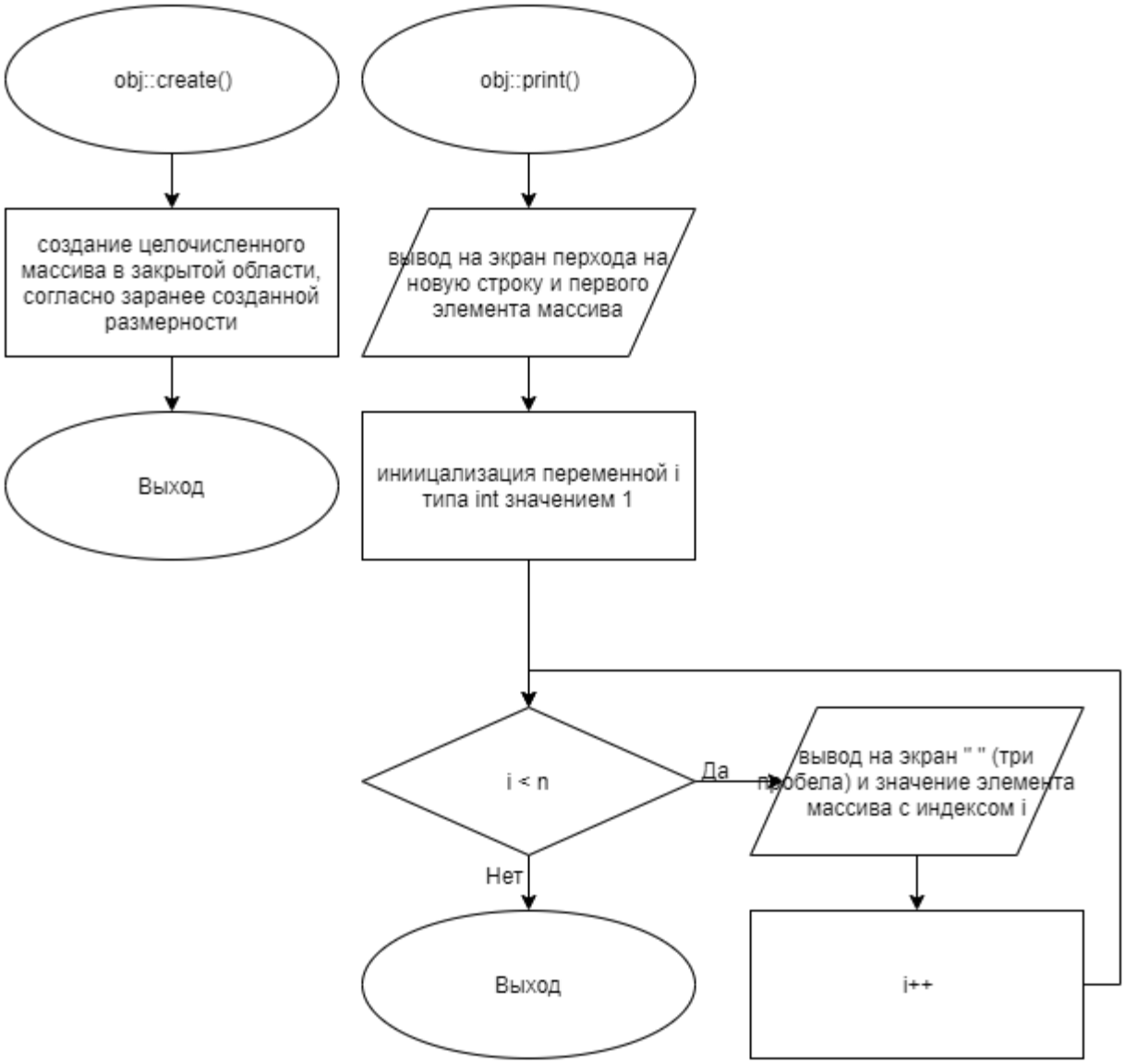

Рисунок 1 – Блок-схема алгоритма

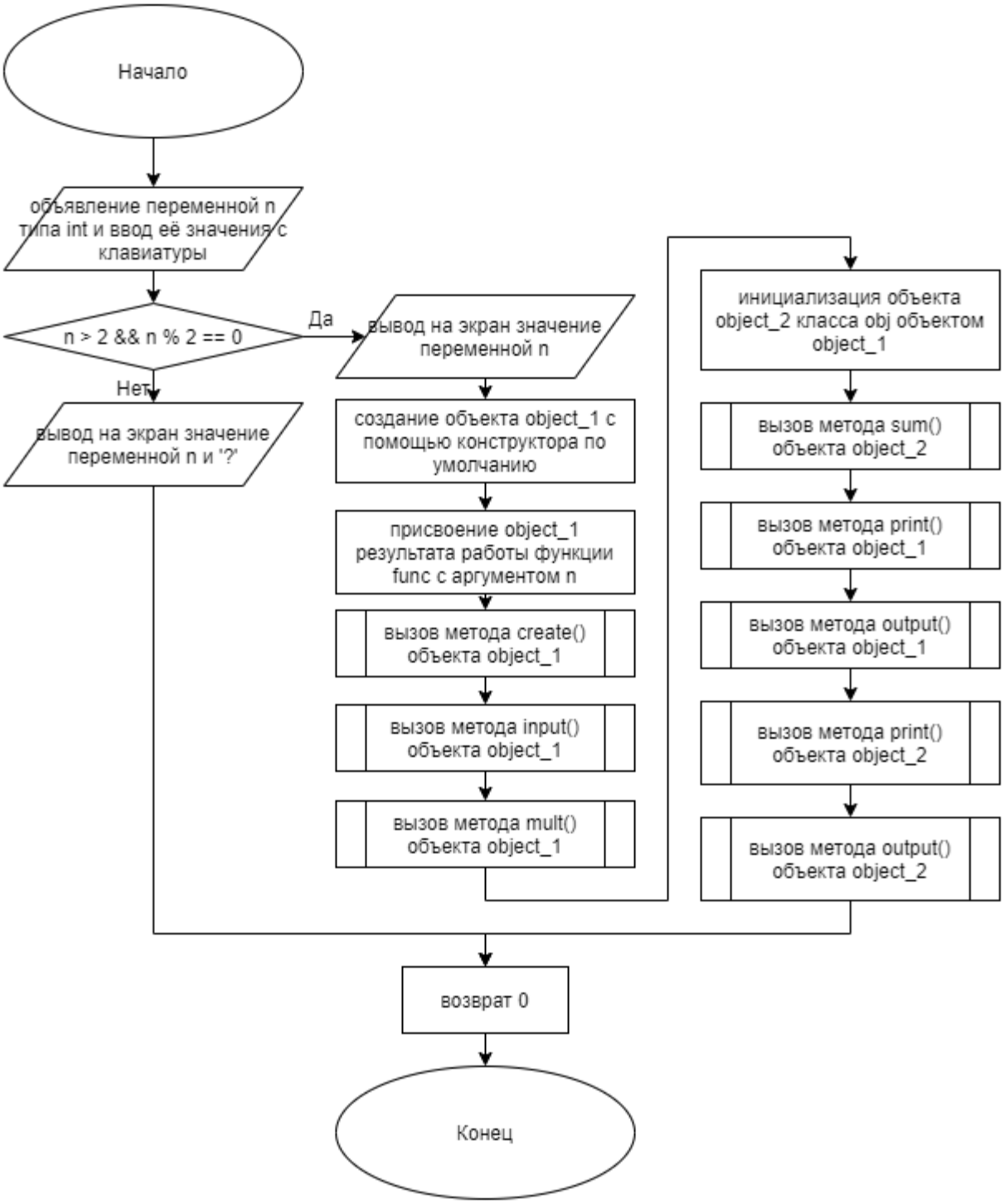

Рисунок 2 – Блок-схема алгоритма

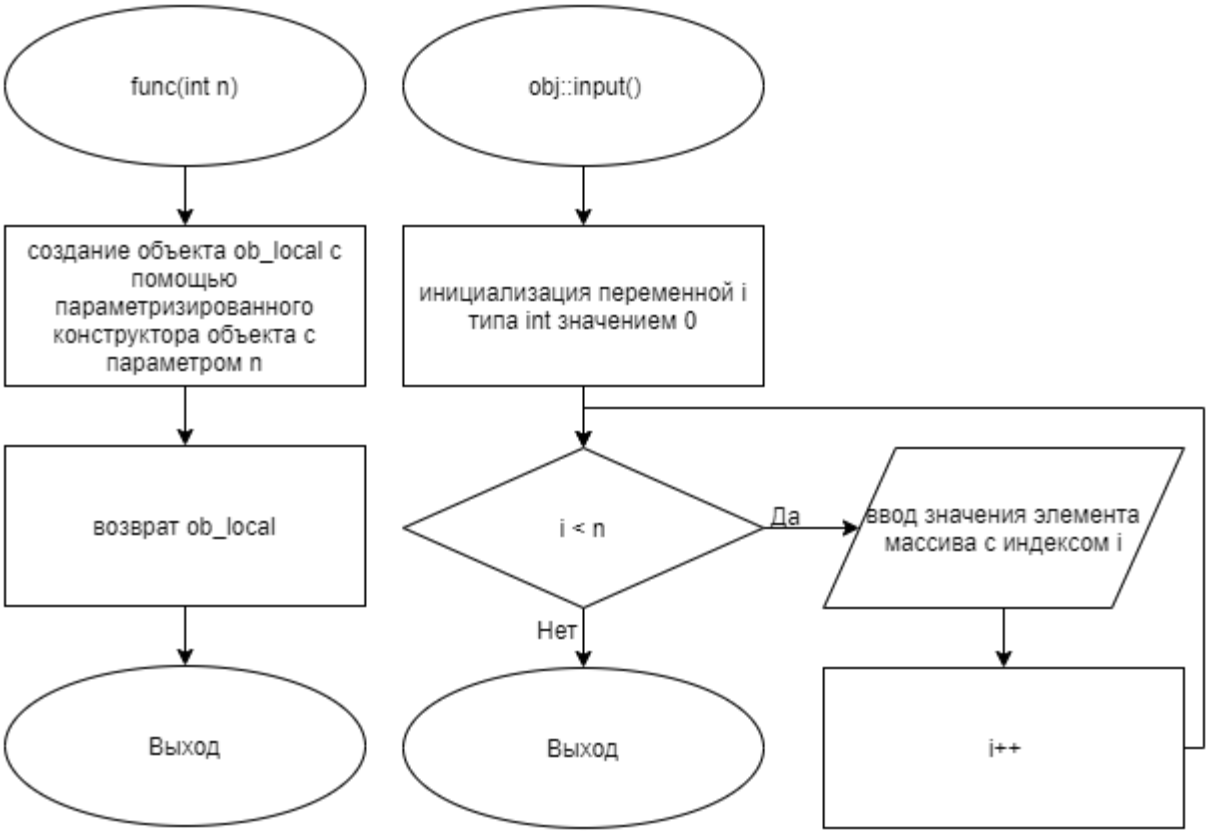

Рисунок 3 – Блок-схема алгоритма

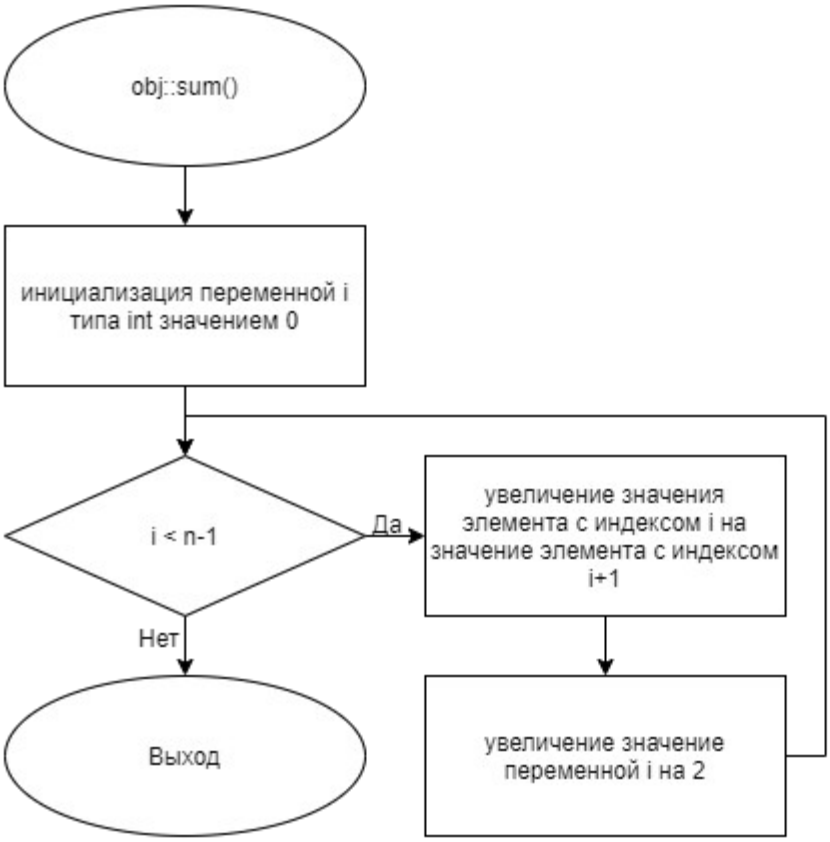

Рисунок 4 – Блок-схема алгоритма

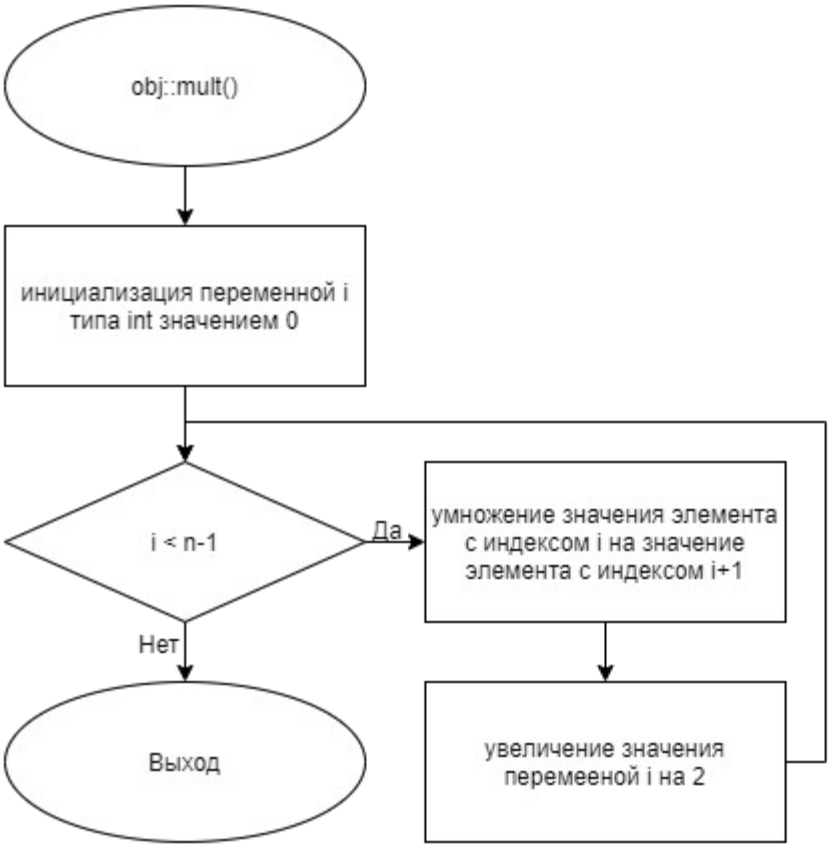

Рисунок 5 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл main.cpp

Листинг 1 – main.cpp

```
#include <stdlib.h>
#include <stdio.h>
#include <iostream>
#include "obj.h"

obj func(int n)
{
   obj ob_local(n);
   return ob_local;
}

int main()
{
   int n; std::cin >> n;
   if (n > 2 && n % 2 == 0)
   {
      std::cout << n;
      obj object_1;
      object_1 = func(n);
      object_1.create();
      object_1.input();
      object_1.mult();
      obj object_2 = object_1;
      object_2.sum();
      object_1.print();
      std::cout << std::endl << object_1.output();
      object_2.print();
      std::cout << std::endl << object_2.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::create() { //new
   matrix = new int [n];
}
void obj::input() {
   for (int i = 0; i < n; i++)
      std::cin >> matrix[i];
}
void obj::sum() {
   for (int i = 0; i < n-1; i+=2)
      matrix[i] += matrix[i+1];
}
void obj::mult() {
   for (int i = 0; i < n-1; i+=2)
      matrix[i] *= matrix[i+1];
}
int obj::output() {
   int sum = 0;
   for (int i = 0; i < n; i++)
      sum += matrix[i];
   return sum;
}
void obj::print() { //new
   std::cout << std::endl << matrix[0];
   for (int i = 1; i < n; i++)
      std::cout << "   " << matrix[i];
}
```

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 create(); //new
   void input();
   void sum();
   void mult();
   int output();
   void print(); //new
};

#endif
```

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
4 1 2 3 4	4 Default constructor Constructor set Destructor Copy constructor 2 2 12 4 20 4 2 16 4 26 Destructor Destructor	4 Default constructor Constructor set Destructor Copy constructor 2 2 12 4 20 4 2 16 4 26 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-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).