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:

- стандартный поток ввода и вывода cin, cout.

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 Class() класса Class

Функционал: конструктор по умолчанию.

Параметры: нет.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 1.

Таблица 1 – Алгоритм метода Class() класса Class

№	Предикат	Действия	№ перехода
1		вывод сообщения	∅

3.2 Алгоритм метода Class(int size) класса Class

Функционал: конструктор с параметром размера массива.

Параметры: нет.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 2.

Таблица 2 – Алгоритм метода Class(int size) класса Class

№	Предикат	Действия	№ перехода
1		задание размера массива	2
2		вывод сообщения	∅

3.3 Алгоритм метода Class(const Class& obj) класса Class

Функционал: конструктор копирования.

Параметры: нет.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода Class(const Class& obj) класса Class

№	Предикат	Действия	№ перехода
1		приравнивание теущего объекта к параметру	∅

3.4 Алгоритм деструктора класса Class

Функционал: деструктор.

Параметры: нет.

Алгоритм деструктора представлен в таблице 4.

Таблица 4 – Алгоритм деструктора класса Class

№	Предикат	Действия	№ перехода
1		удаление масива	2
2		вывод сообщенния	∅

3.5 Алгоритм метода Createarrey класса Class

Функционал: создание массива в закрытом поле.

Параметры: нет.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода Createarrey класса Class

№	Предикат	Действия	№ перехода
1		изменение размера массива	∅

3.6 Алгоритм метода push_back класса Class

Функционал: ввод элеметов для массива.

Параметры: нет.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 6.

Таблица 6 – Алгоритм метода push_back класса Class

№	Предикат	Действия	№ перехода
1		ввод значений и присвоение к элементам массива	∅

3.7 Алгоритм метода method1 класса Class

Функционал: первый метод.

Параметры: нет.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода method1 класса Class

№	Предикат	Действия	№ перехода
1		сумирование значений каждой пары	∅

3.8 Алгоритм метода method2 класса Class

Функционал: второй метод.

Параметры: нет.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 8.

Таблица 8 – Алгоритм метода method2 класса Class

№	Предикат	Действия	№ перехода
1		произведение каждой пары массива	∅

3.9 Алгоритм метода sum класса Class

Функционал: суммирование всех элемнтов массива.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 9.

Таблица 9 – Алгоритм метода sum класса Class

№	Предикат	Действия	№ перехода
1		суммирование всех элемнтов массива	∅

3.10 Алгоритм метода print класса Class

Функционал: вывод всех элемнтов массива.

Параметры: нет.

Возвращаемое значение: none.

Алгоритм метода представлен в таблице 10.

Таблица 10 – Алгоритм метода print класса Class

№	Предикат	Действия	№ перехода
1		вывод всех элемнтов массива	∅

3.11 Алгоритм функции func

Функционал: функция, которая создает объект Class.

Параметры: нет.

Возвращаемое значение: Class.

Алгоритм функции представлен в таблице 11.

Таблица 11 – Алгоритм функции func

№	Предикат	Действия	№ перехода
1		сохздание объкта	2
2		возврат объекта	∅

3.12 Алгоритм функции main

Функционал: основная фнкция.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм функции представлен в таблице 12.

Таблица 12 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		ввод размерности массива	2
2	размерность массива некорректная	вывод сообщения и завершить работу алгоритма	∅
		вывод значения размерности массива	3
3		создание первого олбъекта	4
4		присвоение первому объекту результата раюоты фнкции func с аргкментом, содержащим значение размертности	5
5		для первого объекта вызов метода создание	6

№	Предикат	Действия	№ перехода
		массива	
6		для первогго объекта вызов метода ввода данных массива	7
7		для первого объекта вызов метода 2	8
8		инициализация второго бъекта первым объектом	9
9		вызов метожа 1 для второго объекта	10
10		вызов содержимого массива первого объекта	11
11		вывод суммы элементов масива виорого объекта	12
12		вывод суммы элементов массива вторг ообъекта	13
13		ввод суммаы элементов массива вторго объекта	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

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

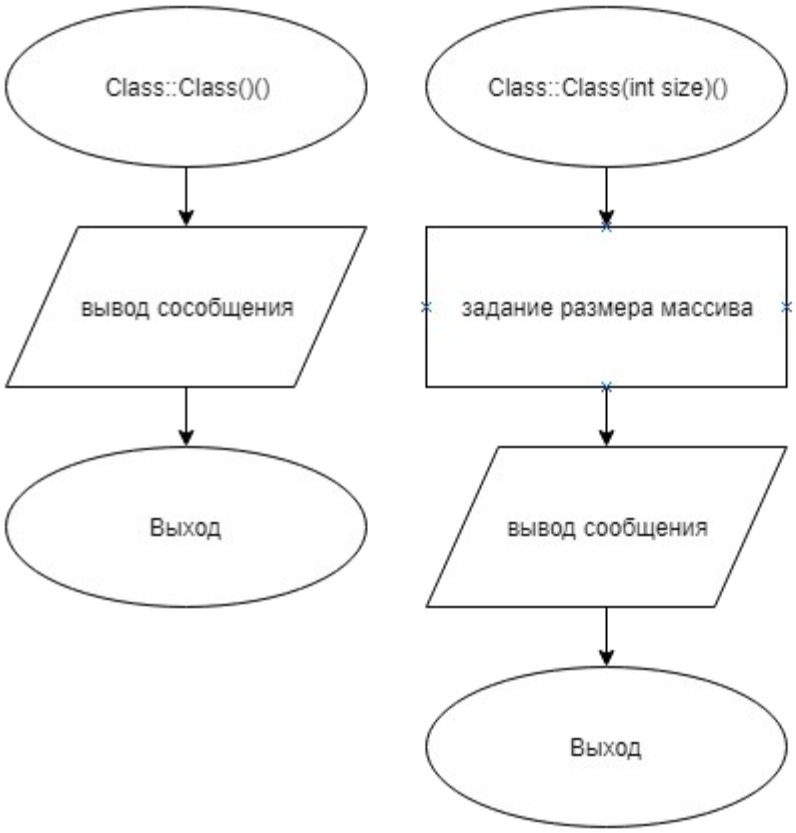

Рисунок 1 – Блок-схема алгоритма

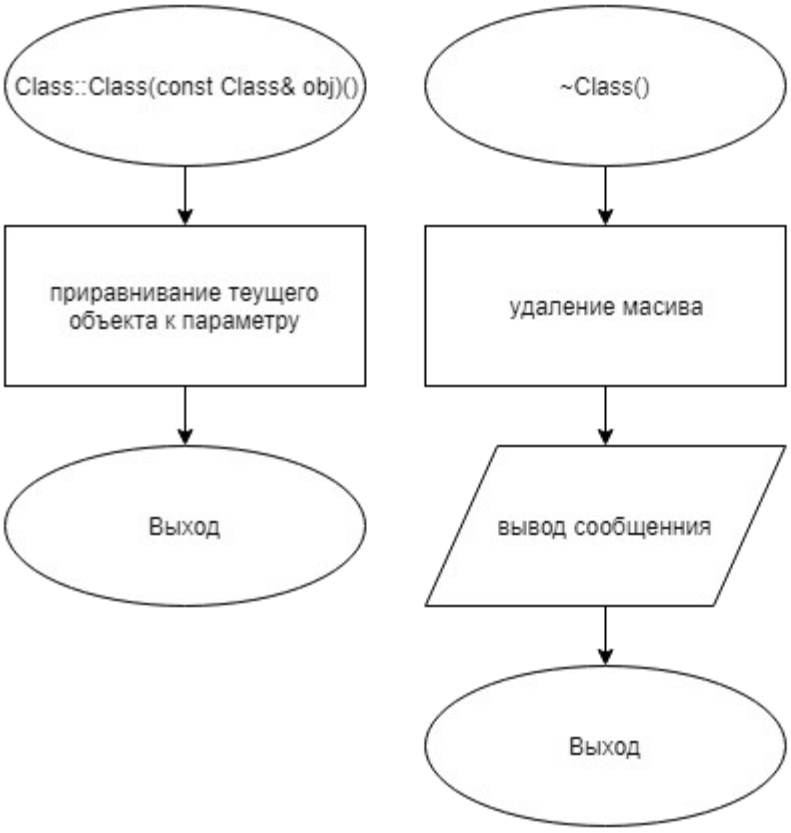

Рисунок 2 – Блок-схема алгоритма

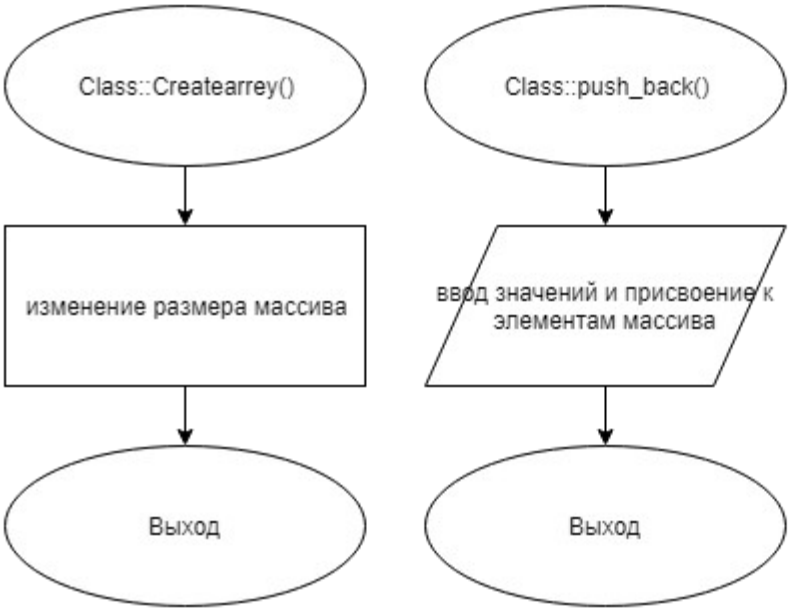

Рисунок 3 – Блок-схема алгоритма

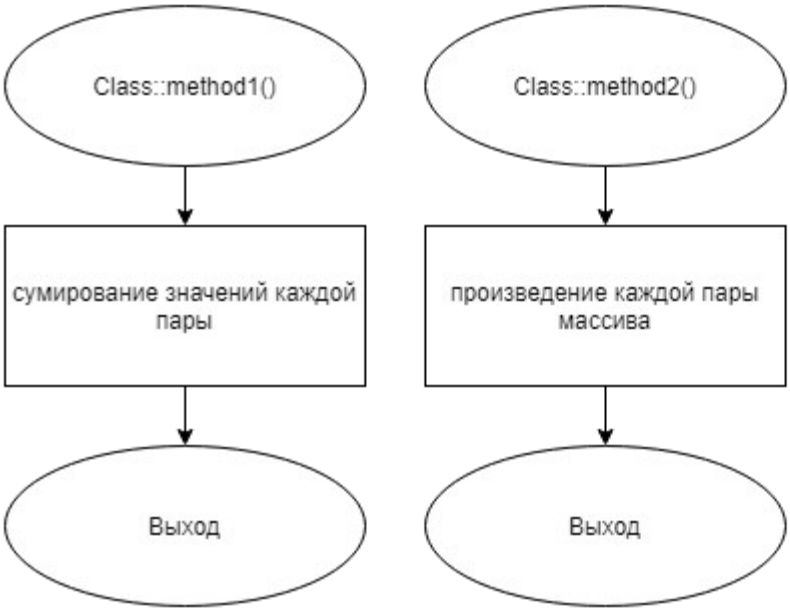

Рисунок 4 – Блок-схема алгоритма

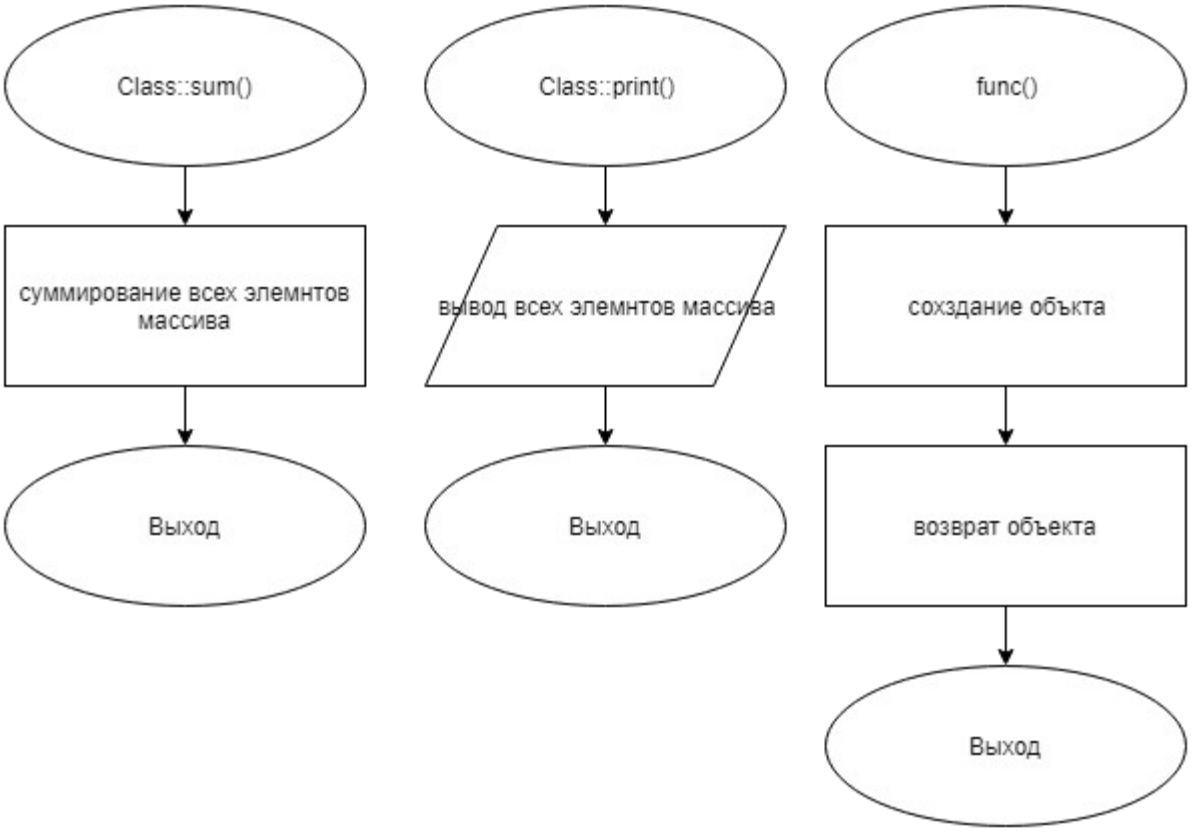

Рисунок 5 – Блок-схема алгоритма

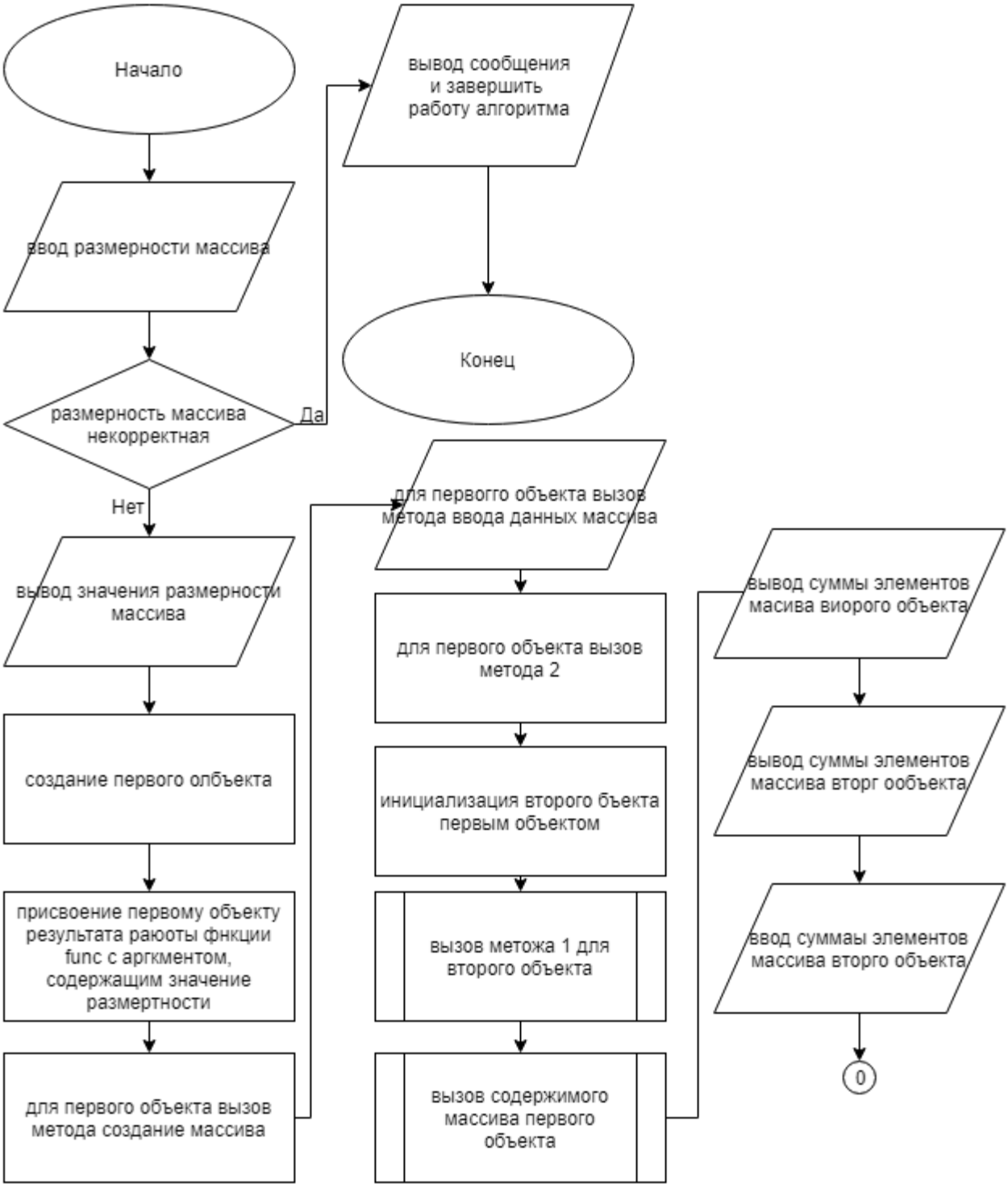

Рисунок 6 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл Class.cpp

Листинг 1 – Class.cpp

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

Class::Class(){cout << "Default constructor\n";}
Class::Class(int n)
{
   size = n;
   cout << "Constructor set";
}

Class::Class(const Class& obj)
{
   size = obj.size;
   array = new int[size];
   for (int i = 0; i < size; i++)
   {
      array[i] = obj.array[i];
   }
   cout << "\nCopy constructor\n";
}

Class::~Class()
{
   delete [] array;
   cout << "\nDestructor";
}

void Class::createarray()
{
   array = new int[size];
}

void Class::push_back()
{
   for (int i = 0; i < size; i++)
   {
      int a= 0;
      while(!(cin >> a))
```

```
            {
               cin.clear();
               cin.ignore(size, '\n');
            }
            array[i] = a;
         }
      }

      void Class::method1()
      {
         for (int i = 0; i < size - 1; i +=2)
         {
            array[i] += array[i + 1];
         }
      }
      void Class::method2()
      {
         for (int i = 0; i < size - 1; i +=2)
         {
            array[i] *= array[i + 1];
         }
      }

      int Class::sum()
      {
         int result = 0;
         for (int i = 0; i < size; i++)
         {
            result += array[i];
         }
         return result;
      }

      void Class::print()
      {
         for(int i = 0; i < size - 1; i++)
         {
            cout << array[i] << "   ";
         }
         cout << array[size - 1];
         cout << endl;
      }
```

5.2 Файл Class.h

Листинг 2 – Class.h

```
      #ifndef __CLASS__H
      #define __CLASS__H
      #include <iostream>
```

```
#include <iomanip>
using namespace std;

class Class
{
private:
   int* array = nullptr;
   int size;
public:
   Class();
   Class(int n);
   Class(const Class& obj);
   ~Class();

   void createarray();
   void push_back();
   void method1();
   void method2();
   int sum();
   void print();
};
#endif
```

5.3 Файл main.cpp

Листинг 3 – main.cpp

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

Class func(int size)
{
   Class object(size);
   return object;
}
int main()
{
   int s1;
   cin >> s1;
   if ((s1 > 2) && (s1 % 2 == 0))
   {
      cout << s1 << endl;
      Class obj1;
      obj1 = func(s1);
      obj1.createarray();
      obj1.push_back();
      obj1.method2();
```

```
            Class obj2(obj1);
            obj2.method1();
            obj1.print();
            cout << obj1.sum() << endl;
            obj2.print();
            cout << obj2.sum();
         }
         else
         {
            cout << s1 << "?";
         }
         return(0);
      }
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 13.

Таблица 13 – 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

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).