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 classname предназначен для ;
- функция function для в качестве параметра получает объект по значению. Функция вызывает метод 2, далее выводит сумму элементов массива с новой строки.;
- функция assert для функция сообщения об ошибке;
- cin/cout - объекты стандартного потока ввода/вывода;
- if..else - условный оператор;
- for - оператор цикла со счётчиком.

Класс classname:

- свойства/поля:
  - поле массив:
    - наименование — a;
    - тип — vector<int>;
    - модификатор доступа — private;
  - поле длина массива:
    - наименование — len_a;
    - тип — int;
    - модификатор доступа — private;
- функционал:
  - метод classname — конструктор по умолчанию;
  - метод classname — параметризированный констркутор;
  - метод classname — конструктор копии;
  - метод ~classname — стандартный дестркутор;
  - метод fill_a — ввод массива;

- метод sum_a — возвращает сумму элементов массива;
- метод half_sum — суммирует значения очередной пары элементов и сумму присваивает первому элементу пары. Далее суммирует элементы полученного массива и возвращает это значение;
- метод half_product — умножает значения очередной пары элементов и результат присваивает первому элементу пары. Далее суммирует элементы полученного массива и возвращает это значение.

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 classname

Функционал: конструктор по умолчанию.

Параметры: нет.

Алгоритм конструктора представлен в таблице 1.

Таблица 1 – Алгоритм конструктора класса classname

№	Предикат	Действия	№ перехода
1		вывод "\nDefault constructor"	∅

3.2 Алгоритм деструктора класса classname

Функционал: стандартный дестркутор.

Параметры: нет.

Алгоритм деструктора представлен в таблице 2.

Таблица 2 – Алгоритм деструктора класса classname

№	Предикат	Действия	№ перехода
1		вывод "\nDestructor"	∅

3.3 Алгоритм метода fill_a класса classname

Функционал: ввод массива.

Параметры: нет.

Возвращаемое значение: void.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода fill_a класса classname

№	Предикат	Действия	№ перехода
1		int n = this->len_a, x, i = 0	2
2	i < n	ввод x	3
			∅
3		this->a.push_back(x)	4
4		++i	2

3.4 Алгоритм конструктора класса classname

Функционал: параметризированный конструктор.

Параметры: int n.

Алгоритм конструктора представлен в таблице 4.

Таблица 4 – Алгоритм конструктора класса classname

№	Предикат	Действия	№ перехода
1		assert(n>2 && n%2==0)	2
2		this->len_a = n	3
3		вывод "\nConstructor set"	∅

3.5 Алгоритм конструктора класса classname

Функционал: конструктор копии.

Параметры: const classname & obj.

Алгоритм конструктора представлен в таблице 5.

Таблица 5 – Алгоритм конструктора класса classname

№	Предикат	Действия	№ перехода
1		len_a = obj.len_a	2
2		a = obj.a	3
3		вывод "\nCopy constructor"	∅

3.6 Алгоритм метода sum_a класса classname

Функционал: возвращает сумму элементов массива.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 6.

Таблица 6 – Алгоритм метода sum_a класса classname

№	Предикат	Действия	№ перехода
1		int n = this->len_a, summa = 0, i = 0	2
2	i < n	summa += this->a[i]	3
			4
3		++i	2
4		возврат summa	∅

3.7 Алгоритм метода half_sum класса classname

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

Параметры: нет.

Возвращаемое значение: int.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода half_sum класса classname

№	Предикат	Действия	№ перехода
1		int n = this->len_a, i = 0	2
2	i < n - 1	this->a[i] = this->a[i] + this->a[i+1];	3
			4
3		++i	2
4		return this->sum_a()	∅

3.8 Алгоритм метода half_product класса classname

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

Параметры: нет.

Возвращаемое значение: int.

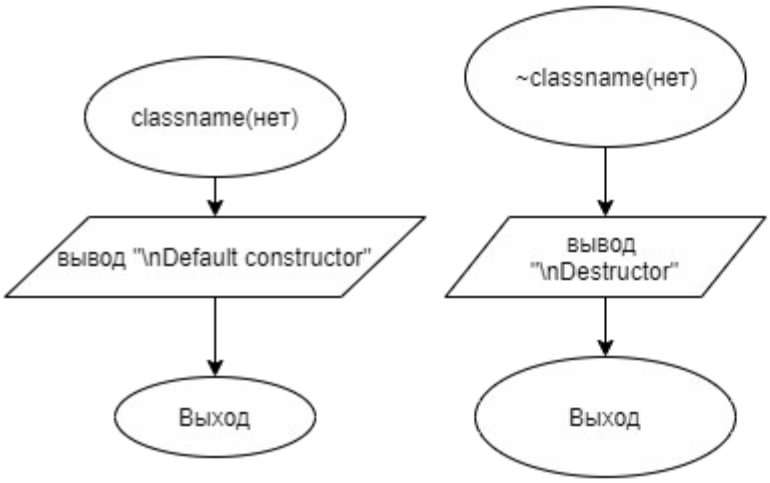
Алгоритм метода представлен в таблице 8.

Таблица 8 – Алгоритм метода half_product класса classname

№	Предикат	Действия	№ перехода
1		int n = this->len_a, i = 0	2
2	i < n - 1	this->a[i] = this->a[i] * this->a[i+1];	3
			4
3		i += 2	2
4		return this->sum_a()	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-4.



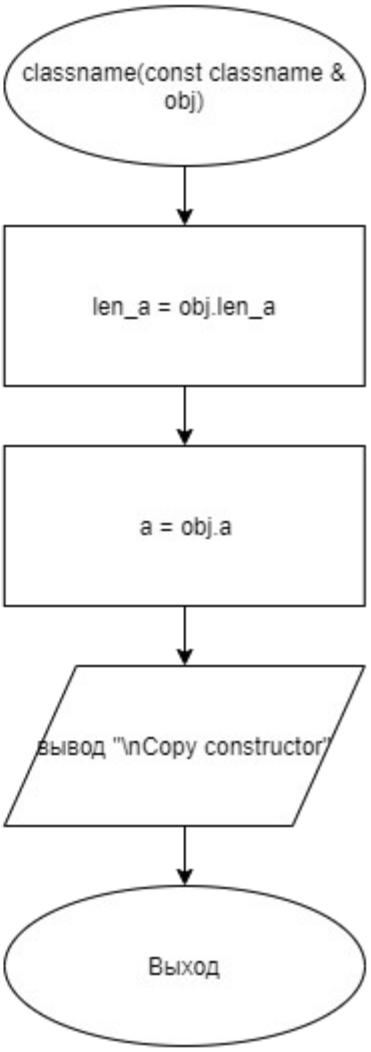


Рисунок 1 – Блок-схема алгоритма

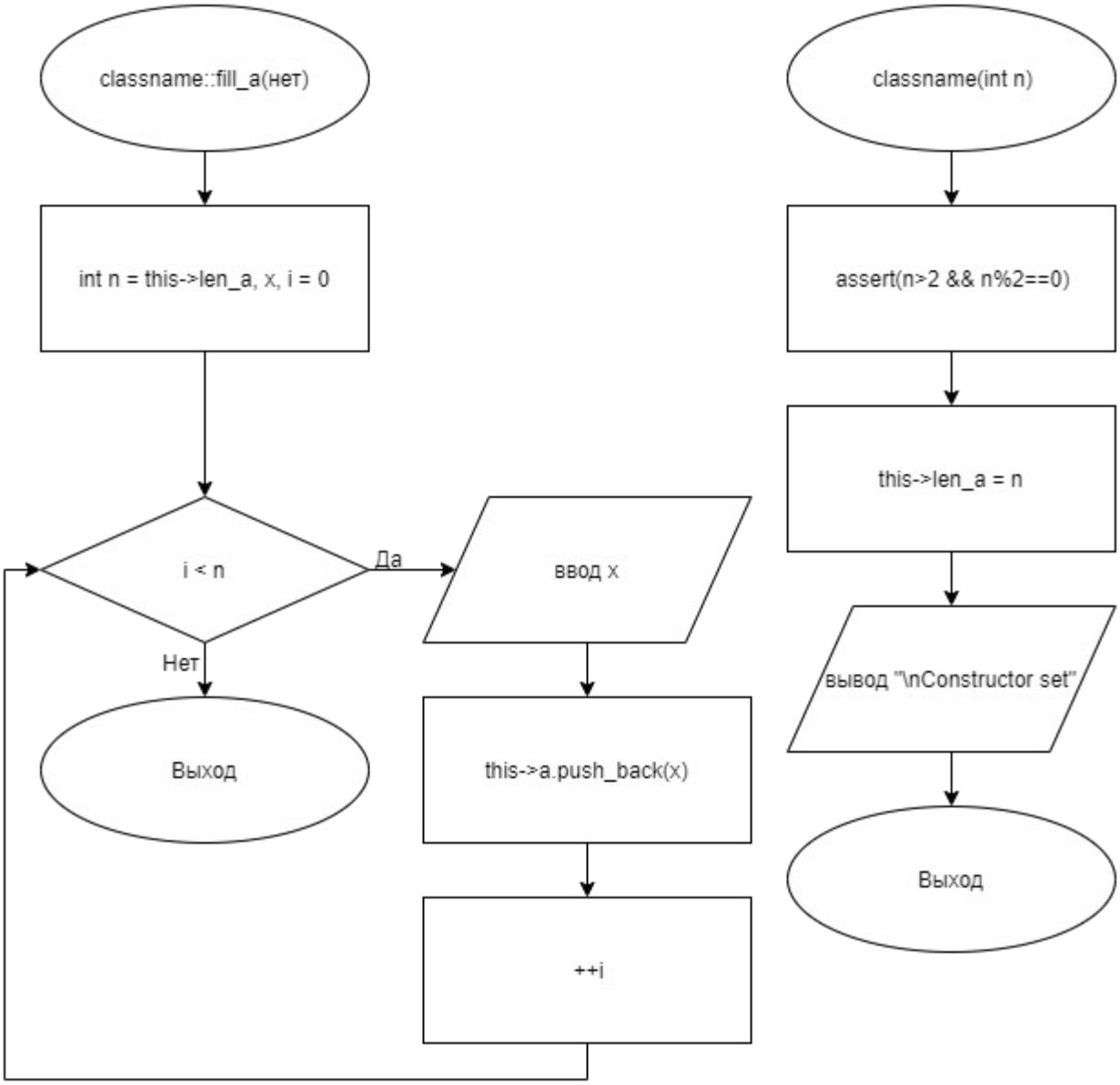


Рисунок 2 – Блок-схема алгоритма

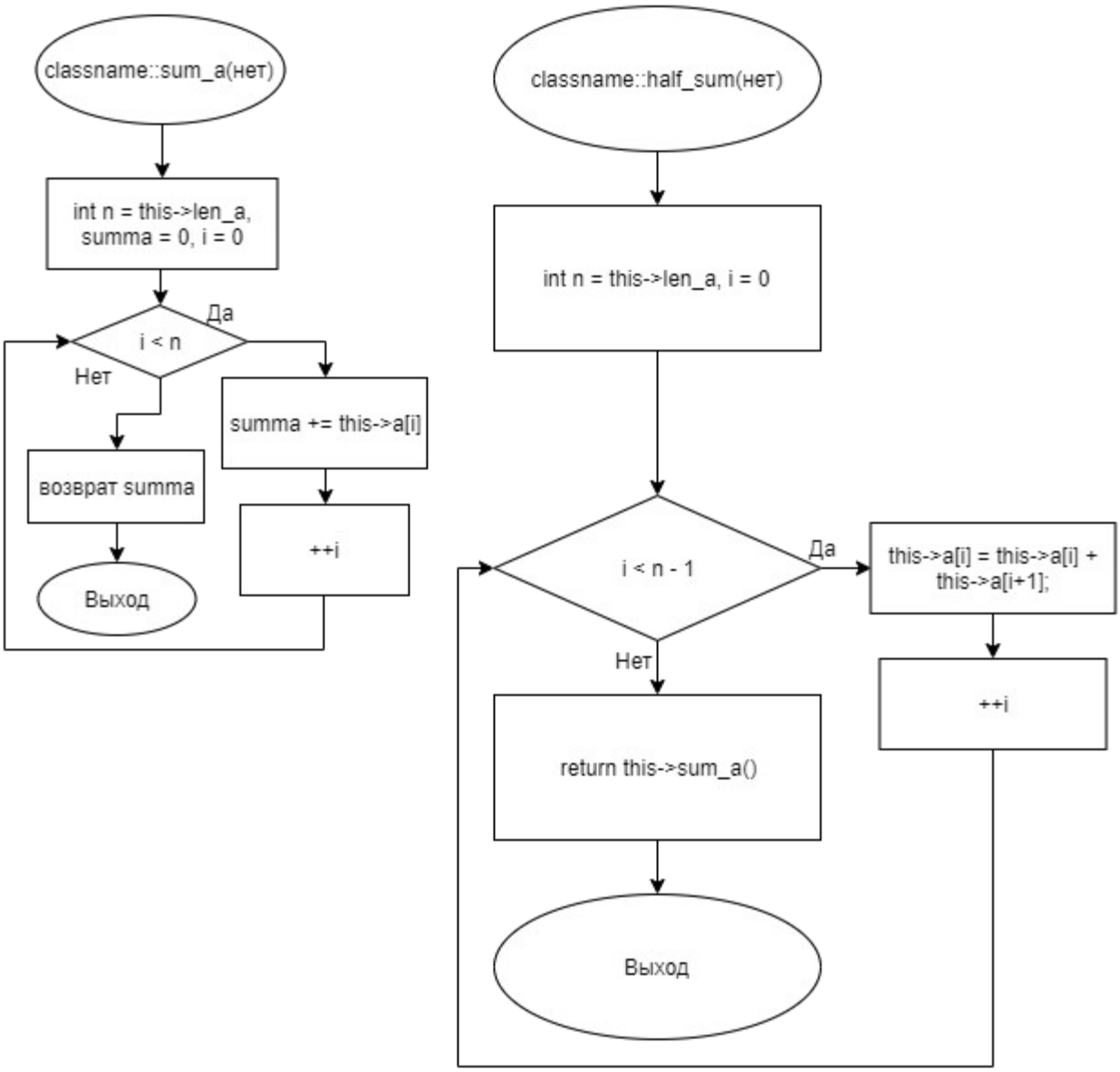


Рисунок 3 – Блок-схема алгоритма

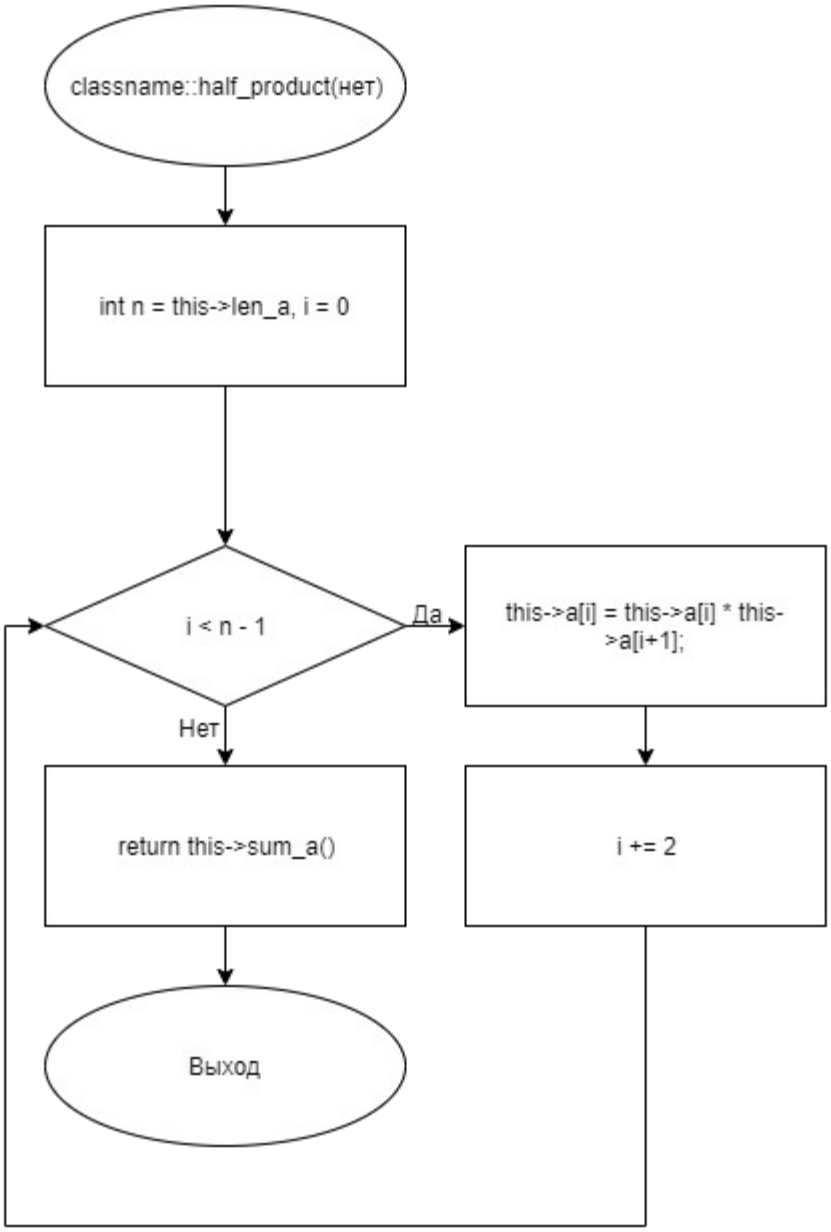


Рисунок 4 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл classname.cpp

Листинг 1 – classname.cpp

```
#include "classname.h"
#include <iostream>
#include <cassert>

classname::classname(){
   std::cout << "\nDefault constructor";
};

classname::classname(int n){
   assert(n>2 && n%2==0);
   this->len_a = n;
   std::cout << "\nConstructor set";
};

classname::~classname(){
   std::cout << "\nDestructor";
};

void classname::fill_a(){
   int n = this->len_a, x;
   for(int i=0;i<n;++i){
      std::cin>>x;
      this->a.push_back(x);

   }

};

int classname::sum_a(){
   int n = this->len_a, summa = 0;
   for(int i=0;i<n;++i)
      summa+=this->a[i];
   return summa;
};

int classname::half_sum(){
   int n = this->len_a;
   for(int i=0;i < n - 1;i+=2)
      this->a[i] = this->a[i] + this->a[i+1];
```

```
   return this->sum_a();
};

int classname::half_product(){
   int n = this->len_a;
   for(int i=0;i < n - 1;i+=2)
      this->a[i] = this->a[i] * this->a[i+1];
   return this->sum_a();
};

classname::classname(const classname & obj){
   len_a = obj.len_a;
   a = obj.a;
   std::cout<<"\nCopy constructor";
};
```

5.2 Файл classname.h

Листинг 2 – classname.h

```
#ifndef __CLASSNAME__H
#define __CLASSNAME__H
#include <vector>

class classname{
   std::vector<int>a;
   int len_a;
public:
   classname();
   classname(int n);
   classname(const classname & obj);
   ~classname();
   void fill_a();
   int sum_a();
   int half_sum();
   int half_product();
};

#endif
```

5.3 Файл main.cpp

Листинг 3 – main.cpp

```
//#include <stdlib.h>
```

```
//#include <stdio.h>
#include <iostream>
#include "classname.h"

void function(classname obj){
   std::cout<<'\n'<<obj.half_product();
}

int32_t main()
{
   int n; std::cin>>n;
   if(!(n>2 && n%2==0)){
      std::cout<<n<<'?';
      return 0;
   }
   std::cout << n;
   classname obj(n);
   obj.fill_a();
   function(obj);
   std::cout<<'\n'<<obj.half_sum();
   return(0);
}
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 9.

Таблица 9 – 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
2	2?	2?
7	7?	7?

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).