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:

3. Ввод размерности массива.
4. Если размерность массива некорректная, вывод сообщения и завершить работу алгоритма.
5. Вывод значения размерности массива.
6. Создание первого объекта.
7. Присвоение первому объекту результата работы функции func с аргументом, содержащим значение размерности массива.
8. Для первого объекта вызов метода создания массива.
9. Для первого объекта вызов метода ввода данных массива.
10. Для первого объекта вызов метода 2.
11. Инициализация второго объекта первым объектом.
12. Вызов метода 1 для второго объекта.
13. Вывод содержимого массива первого объекта.
14. Вывод суммы элементов массива первого объекта.
15. Вывод содержимого массива второго объекта.
16. 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

Операторы ввода/вывода cin/cout

Операторы new/delete

Метод Print

Функция func

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 Print класса MyClass

Функционал: Вывод массива.

Параметры: отсутствуют.

Возвращаемое значение: целый тип.

Алгоритм метода представлен в таблице 1.

Таблица 1 – Алгоритм метода Print класса MyClass

№	Предикат	Действия	№ перехода
1		Вывод всех элементов массива	∅

3.2 Алгоритм функции func

Функционал: Создание объекта с параметризированным конструктором.

Параметры: int, l, длина массива.

Возвращаемое значение: тип класса.

Алгоритм функции представлен в таблице 2.

Таблица 2 – Алгоритм функции func

№	Предикат	Действия	№ перехода
1		Создание объекта с параметризированным конструктором	2
2		Возвращение этого объекта	∅

3.3 Алгоритм функции main

Функционал: основная функция.

Параметры: отсутствуют.

Возвращаемое значение: целый тип.

Алгоритм функции представлен в таблице 3.

Таблица 3 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		Создание/ввод переменной l	2
2	l > 2 && L % 2 == 0	Вывод l	3
		Вывод l?	∅
3		Создание объекта obj_a класса MyClass	4
4		obj_a = func(l)	5
5		Вызов метода Input(), для объекта obj_a	6
6		Вызов метода Met_b(), для объекта obj_a	7
7		Создание объекта obj_b класса MyClass с параметром (obj_a)	8
8		Вызов метода Met_a(), для объекта obj_b	9
9		Вызов метода Print(), для объекта obj_a	10
10		Вывод результата метода Summ(), для объекта obj_a	11
11		Вызов метода Print(), для объекта obj_b	12
12		Вывод результата метода Summ(), для объекта obj_b	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-2.

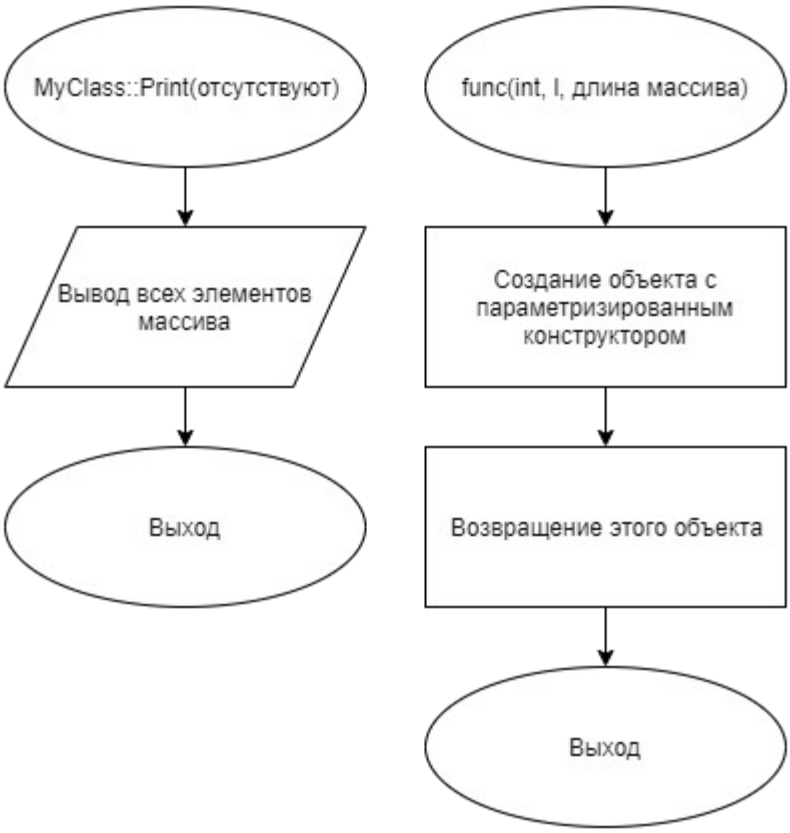

Рисунок 1 – Блок-схема алгоритма

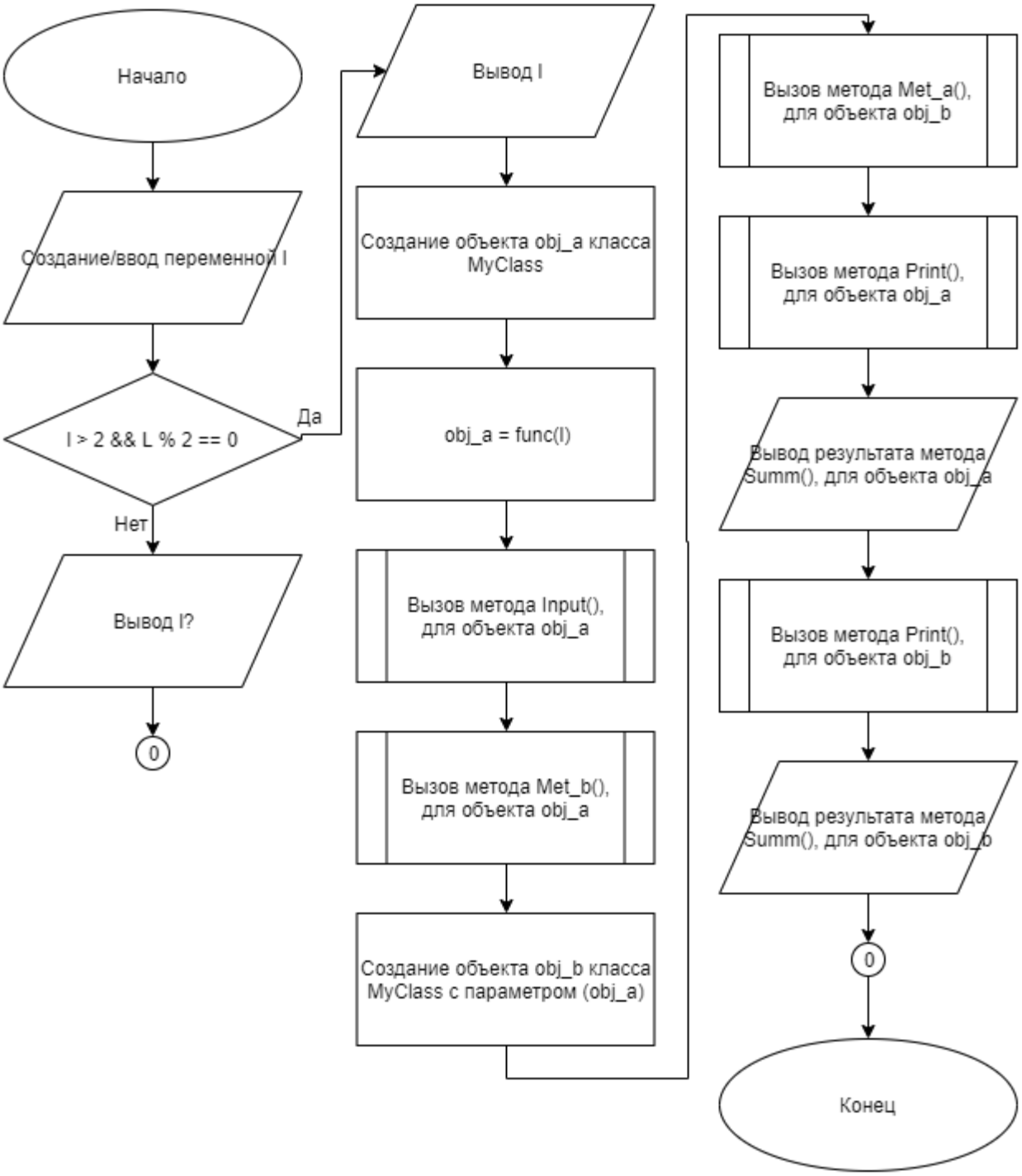

Рисунок 2 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл main.cpp

Листинг 1 – main.cpp

```
#include "MyClass.h"

MyClass func(int l)
{
	MyClass obj_func(l);
	return obj_func;
}

int main()
{
	int l;
	cin >> l;
	cout << l;
	if ((l > 2) && (l % 2 == 0))
	{
		MyClass obj_a;
		obj_a = func(l);
		obj_a.Input();
		obj_a.Met_b();
		MyClass obj_b(obj_a);
		obj_b.Met_a();
		obj_a.Print();
		cout << endl << obj_a.Summ();
		obj_b.Print();
		cout << endl << obj_b.Summ();
	}
	else
	{
		cout << "?";
	}
}
```

5.2 Файл MyClass.cpp

Листинг 2 – MyClass.cpp

```
#include "MyClass.h"
```

```
int dest = 0;

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

MyClass::MyClass(int l)
{
	cout << "\nConstructor set";
	len = l;
	mas = new int[len];
}

MyClass::~MyClass()
{
	cout << "\nDestructor";
	if (mas != nullptr && dest == 3)
	{
		delete []mas;
	}
}

MyClass::MyClass(MyClass& obj)
{
	cout << "\nCopy constructor";
	len = obj.len;
	mas = new int[len];
	for (int i = 0; i < len; i++)
	{
		mas[i] = obj.mas[i];
	}
}

void MyClass::Input()
{
	for (int i = 0; i < len; i++)
	{
		cin >> mas[i];
	}
}

void MyClass::Met_a()
{
	for (int i = 0; i+1 < len; i = i + 2)
	{
		mas[i] = mas[i] + mas[i+1];
	}
}

void MyClass::Met_b()
{
	for (int i = 0; i+1 < len; i = i + 2)
	{
		mas[i] = mas[i] * mas[i+1];
	}
}
```

```
int MyClass::Summ()
{
	int summ = 0;
	for (int i = 0; i < len; i++)
	{
		summ = summ + mas[i];
	}
	return summ;
}

void MyClass::Print()
{
	cout << endl;
	for (int i = 0; i < len; i++)
	{
		cout << mas[i];
		if (i != len-1)
		{
			cout << "   ";
		}
	}
}
```

5.3 Файл MyClass.h

Листинг 3 – MyClass.h

```
#ifndef __MYCLASS_H__
#define __MYCLASS_H__
#include <iostream>
using namespace std;
class MyClass
{
public:
	MyClass();
	MyClass(int l);
	MyClass(MyClass& obj);
	~MyClass();
	void Input();
	void Met_a();
	void Met_b();
	int Summ();
	void Print();
private:
	int *mas, len = 0;
};
#endif
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 4.

Таблица 4 – 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
2	2?	2?
7	7?	7?

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Васильев А.Н. Объектно-ориентированное программирование на C++. Издательство: Наука и Техника. Санкт-Петербург, 2016г. 543 стр.

2. Шилдт Г. C++: базовый курс. 3-е изд. Пер. с англ.. — М.: Вильямс, 2017. — 624 с.

3. Методическое пособие для проведения практических заданий, контрольных и курсовых работ по дисциплине «Объектно-ориентированное программирование» [Электронный ресурс] – URL: https://mirea.aco-avrora.ru/student/files/methodichescoe_posobie_dlya_laboratorny h_rabot_3.pdf (дата обращения 05.05.2021).

4. 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).

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).