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 значений элементов 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 двумя пробелами;

- метод, который возвращает значение указателя на массив из закрытой области;
- метод, который присваивает значение указателя массива из закрытой области.

Назовём класс описания данного объекта cl_obj (для примера, у вас он может называться иначе).

Разработать функцию func, которая имеет один целочисленный параметр, содержащий размерность массива. В функции должен быть реализован алгоритм:

- Инициализация указателя на объект класса cl_obj адресом объекта, созданного с использованием параметризированного конструктора.
- С использованием указателя на объект класса cl_obj вызов метода создания массива.
- С использованием указателя на объект класса cl_obj вызов метода ввода значений элементов массива.
- С использованием указателя на объект класса cl_obj вызов метода 2.
- Возврат указателя на объект класса cl_obj.

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. Объявить первый указатель на объект класса cl_obj.
5. Присвоение первому указателю результата работы функции func с аргументом, содержащим значение размерности массива.
6. С использованием первого указателя вызов метода 1.
7. Инициализация второго указателя на объект класса cl_obj адресом объекта,

созданного с использованием конструктора копии с аргументом первого объекта.

8. С использованием второго указателя вызов метода 2.
9. Вывод содержимого массива первого объекта.
10. Вывод суммы элементов массива первого объекта.
11. Вывод содержимого массива второго объекта.
12. Вывод суммы элементов массива второго объекта.
13. Второму объекту присвоить первый объект.
14. С использованием первого указателя вызов метода 1.
15. Вывод содержимого массива второго объекта.
16. Вывод суммы элементов массива второго объекта.
17. Удалит первый объект.
18. Удалить второй объект.

Добавить в этот алгоритм пункты, которые обеспечат корректное завершение работы программы.

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
Constructor set
Copy constructor
20  5  4  2
31
100  5  8  2
115
100  5  8  2
115
Destructor
Destructor
```

2 МЕТОД РЕШЕНИЯ

Класс Class:

- Функционал:
  - Метод ReturnPtr() - возвращает значение указателя на массив из закрытой области;
  - Метод SetPtr(int* p) - присваивает значение указателя массива из закрытой области.

Функция func(int size) - Инициализация указателя на объект класса cl_obj адресом объекта, созданного с использованием параметризированного конструктора, вызов метода создания массива, вызов метода ввода значений элементов массива, Возврат указателя на объект класса cl_obj.

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 ReturnPtr класса Class

Функционал: возвращает значение указателя на массив из закрытой области.

Параметры: .

Возвращаемое значение: int*.

Алгоритм метода представлен в таблице 1.

Таблица 1 – Алгоритм метода ReturnPtr класса Class

№	Предикат	Действия	№ перехода
1		Возврат значения указателя на массив mas	∅

3.2 Алгоритм метода SetPtr класса Class

Функционал: присваивает значение указателя массива из закрытой области.

Параметры: int* p.

Возвращаемое значение: void.

Алгоритм метода представлен в таблице 2.

Таблица 2 – Алгоритм метода SetPtr класса Class

№	Предикат	Действия	№ перехода
1		mas = p	∅

3.3 Алгоритм функции func

Функционал: Инициализация указателя на объект класса cl_obj адресом объекта, созданного с использованием параметризированного конструктора, вызов метода создания массива, вызов метода ввода значений элементов массива, Возврат указателя на объект класса cl_obj..

Параметры: int size.

Возвращаемое значение: Class*.

Алгоритм функции представлен в таблице 3.

Таблица 3 – Алгоритм функции func

№	Предикат	Действия	№ перехода
1		Инициализация указателя на объект класса cl_obj адресом объекта loc, созданного с использованием параметризированного конструктора с параметром size	2
2		Вызов метода Create() объекта loc	3
3		Вызов метода Input() объекта loc	4
4		Вызов метода Prod() объекта loc	5
5		Возврат указателя на объект loc класса cl_obj	∅

3.4 Алгоритм функции main

Функционал: Основная программа.

Параметры: .

Возвращаемое значение: int - код возврата.

Алгоритм функции представлен в таблице 4.

Таблица 4 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		Объявление целочисленной переменной size	2

№	Предикат	Действия	№ перехода
2		Ввод значения переменной size	3
3	size <= 2 или size нечетное	Вывод значения переменной size со знаком вопроса	∅
		Вывод значения переменной size	4
4		Вывод переноса на новую строку	5
5		Объявления указателя obj1 на объект класса Class	6
6		Присвоение указателю obj1 результата работы функции func с аргументом size	7
7		Вывод переноса на новую строку	8
8		Вызов метода SumPara() объекта obj1	9
9		Инициализация указателя obj2 на объект класса Class адресом объекта, созданного с использованием конструктора копии с аргументом объекта obj1	10
10		Вызов метода Prod() объекта obj2	11
11		Вызов метода Print() объекта obj1	12
12		Вывод переноса на новую строку	13
13		Вывод значения результата вызова метода Sum() объекта obj1	14
14		Вывод переноса на новую строку	15
15		Вызов метода Print() объекта obj2	16
16		Вывод переноса на новую строку	17
17		Вывод значения результата вызова метода Sum() объекта obj2	18

№	Предикат	Действия	№ перехода
18		Вывод переноса на новую строку	19
19		Инициализация указателя p значением результата работы метода ReturnPtr объекта obj2	20
20		*obj2 = *obj1	21
21		Вызов метода SumPara() объекта obj1	22
22		Вызов метода SetPtr(p) объекта obj2	23
23		Вызов метода Print() объекта obj2	24
24		Вывод переноса на новую строку	25
25		Вывод значения результата вызова метода Sum() объекта obj2	26
26		Удаление объекта по адресу указателя obj1 при помощи оператора функции delete	27
27		Удаление объекта по адресу указателя obj2 при помощи оператора функции delete	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-3.

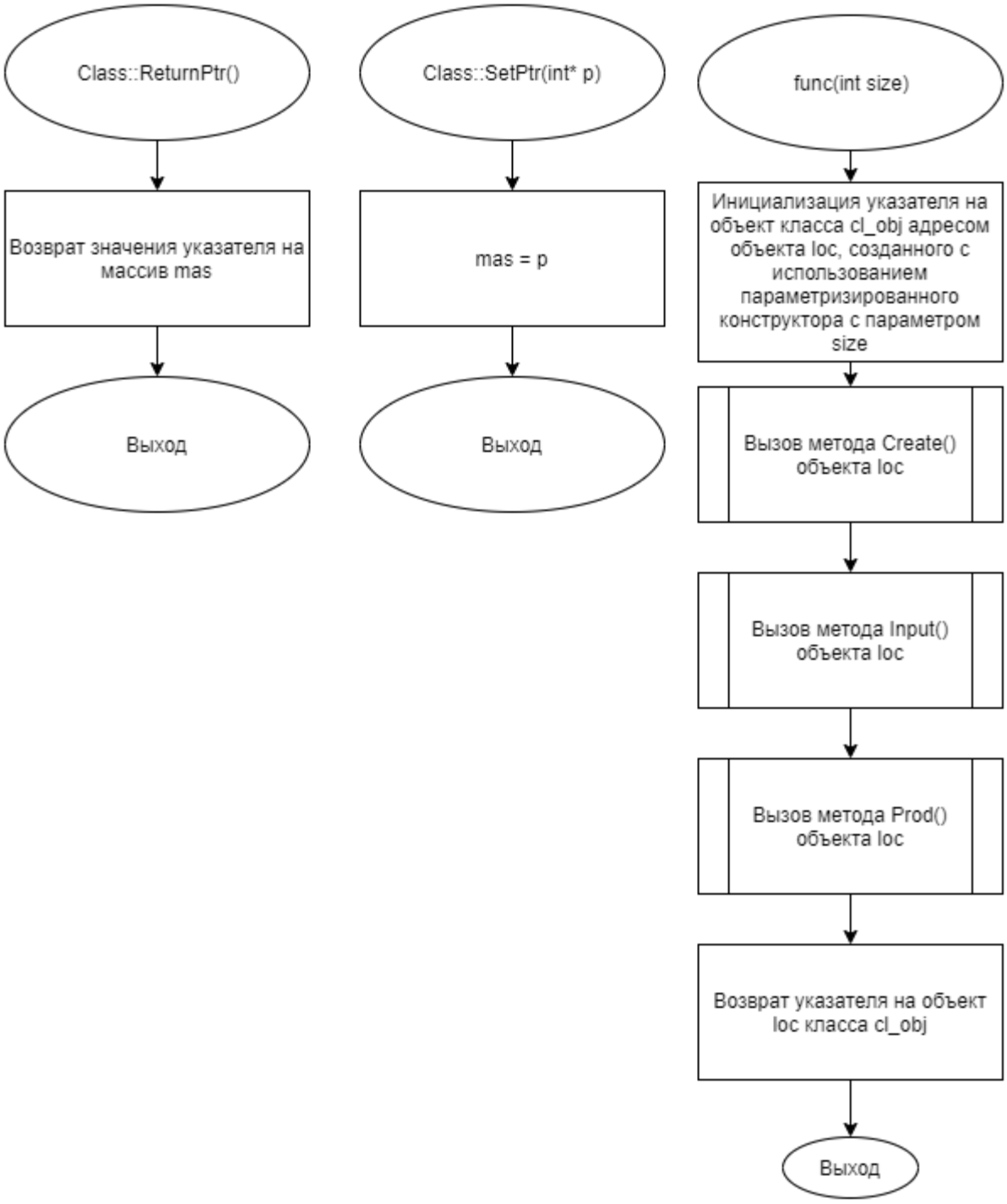

Рисунок 1 – Блок-схема алгоритма

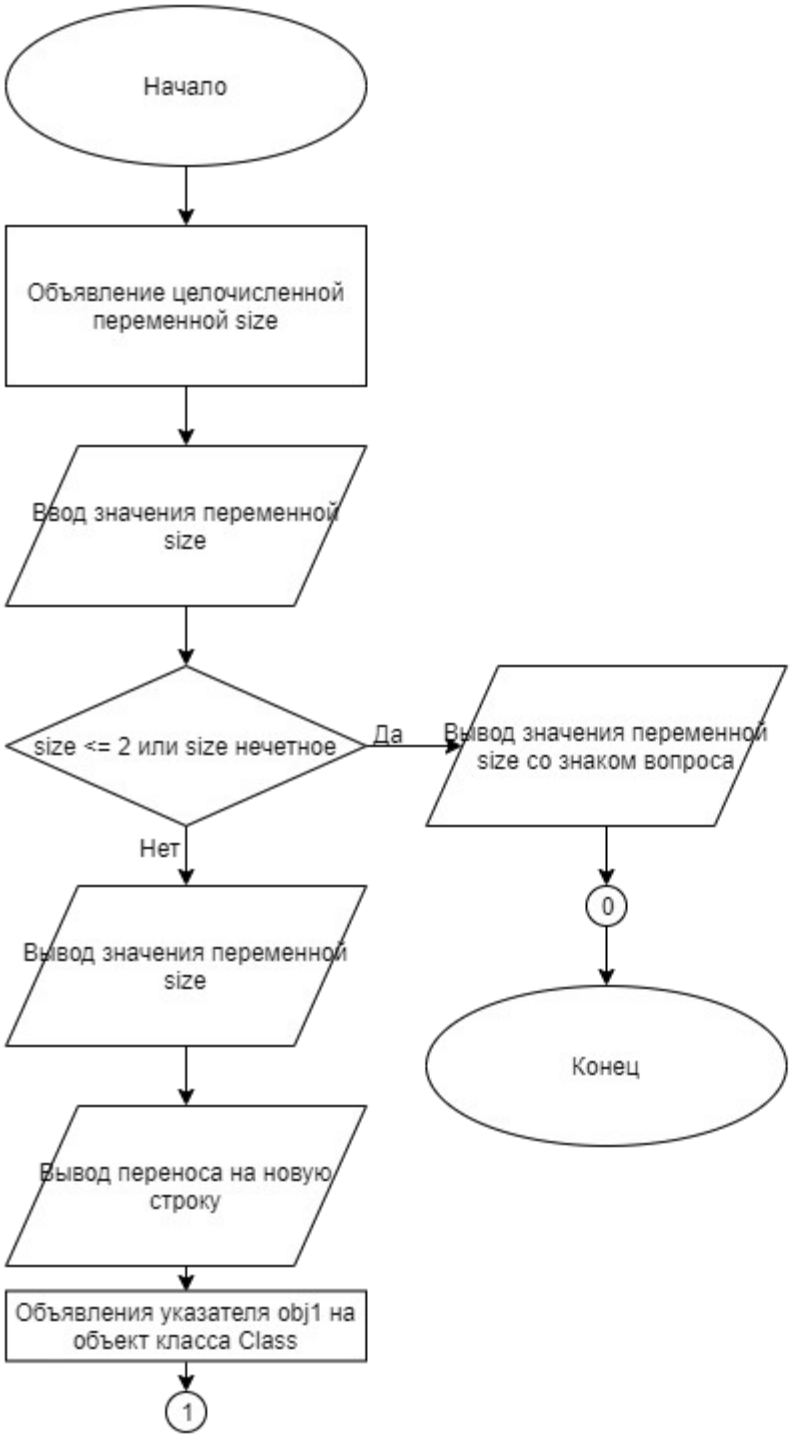

Рисунок 2 – Блок-схема алгоритма

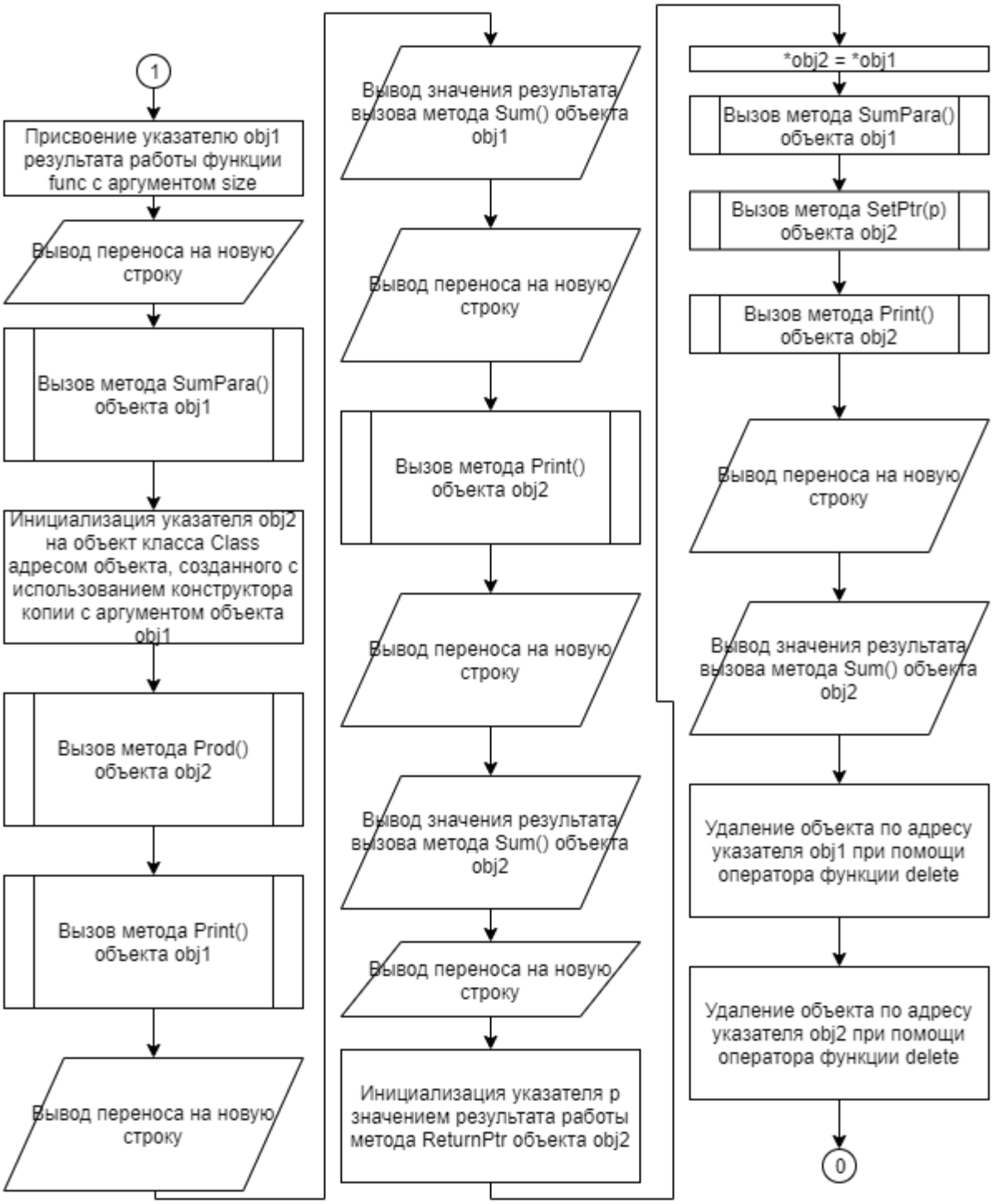

Рисунок 3 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл Class.cpp

Листинг 1 – Class.cpp

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

Class::Class() {
	cout << "Default constructor" << endl;
	mas = nullptr;
}
Class::Class(int n) {
	cout << "Constructor set";
	mas = new int[n];
	this->n = n;
}
Class::Class(const Class& obj) {
	cout << "Copy constructor" << endl;
	n = obj.n;
	mas = new int[n];
	for (int i = 0; i < n; i++) {
		mas[i] = obj.mas[i];
	}
}
Class::~Class() {
	cout << endl << "Destructor";
	if (mas != nullptr) {
		delete[] mas;
	}
}
void Class::Create() {
	mas = new int[n];
}
void Class::Input() {
	int x;
	for (int i = 0; i < n; i++) {
		cin >> x;
		mas[i] = x;
	}
}
void Class::SumPara() {
	for (int i = 0; i < n; i += 2) {
		mas[i] = mas[i] + mas[i + 1];
	}
```

```
}
void Class::Prod() {
	for (int i = 0; i < n; i += 2) {
		mas[i] = mas[i] * mas[i + 1];
	}
}
int Class::Sum() {
	int s = 0;
	for (int i = 0; i < n; i++) {
		s += mas[i];
	}
	return s;
}
void Class::Print() {
	for (int i = 0; i < n; i++) {
		cout << mas[i];
		if (i != n - 1) {
			cout << "  ";
		}
	}
}
int* Class::ReturnPtr() {
	return mas;
}
void Class::SetPtr(int* p) {
	mas = p;
}
```

5.2 Файл Class.h

Листинг 2 – Class.h

```
#ifndef __CLASS_H
#define __CLASS_H
class Class {
private:
	int n;
	int *mas;
public:
	Class();
	Class(int n);
	Class(const Class& obj);
	~Class();
	void Create();
	void Input();
	void SumPara();
	void Prod();
	int Sum();
	void Print();
	int* ReturnPtr();
	void SetPtr(int* p);
};
#endif
```

5.3 Файл main.cpp

Листинг 3 – main.cpp

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

Class* func(int size) {
      Class* loc = new Class(size);
      loc->Create();
      loc->Input();
      loc->Prod();
      return loc;
      delete loc;
}

int main() {
      int size;
      cin >> size;
      if (size <= 2 || size % 2 != 0) {
            cout << size << "?";
            return 0;
      }
      cout << size;
      cout << endl;
      Class* obj1; // 4
      obj1 = func(size); // 5
      cout << endl;
      obj1->SumPara(); // 6
      Class* obj2 = new Class(*obj1); // 7
      obj2->Prod(); // 8
      obj1->Print(); // 9
      cout << endl;
      cout << obj1->Sum(); // 10
      cout << endl;
      obj2->Print(); // 11
      cout << endl;
      cout << obj2->Sum(); // 12
      cout << endl;

      int* p = obj2->ReturnPtr();
      *obj2 = *obj1; // 13

      obj1->SumPara(); // 14
      obj2->SetPtr(p);
      obj2->Print(); // 15
      cout << endl;
      cout << obj2->Sum(); // 16
      delete obj1;
      delete obj2;
}
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 5.

Таблица 5 – Результат тестирования программы

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
4 3 5 1 2	4 Constructor set Copy constructor 20 5 4 2 31 100 5 8 2 115 100 5 8 2 115 Destructor Destructor	4 Constructor set Copy constructor 20 5 4 2 31 100 5 8 2 115 100 5 8 2 115 Destructor Destructor

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).