Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ.....5
1.1 Описание входных данных.....7
1.2 Описание выходных данных.....8
2 МЕТОД РЕШЕНИЯ.....10
3 ОПИСАНИЕ АЛГОРИТМОВ.....11
3.1 Алгоритм метода PMassGet класса MeineKlasse.....11
3.2 Алгоритм метода PMassSet класса MeineKlasse.....11
3.3 Алгоритм функции main.....12
3.4 Алгоритм функции creationFunction.....13
4 БЛОК-СХЕМЫ АЛГОРИТМОВ.....14
5 КОД ПРОГРАММЫ.....17
5.1 Файл main.cpp.....17
5.2 Файл MeineKlasse.cpp.....18
5.3 Файл MeineKlasse.h.....19
6 ТЕСТИРОВАНИЕ.....21
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....22

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:

1. Инициализация указателя на объект класса cl_obj адресом объекта, созданного с использованием параметризированного конструктора.
2. С использованием указателя на объект класса cl_obj вызов метода создания массива.
3. С использованием указателя на объект класса cl_obj вызов метода ввода значений элементов массива.
4. С использованием указателя на объект класса cl_obj вызов метода 2.
5. 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

Для решения задачи используется:

- объект temp_obj класса MeineKlasse предназначен для временного хранения значения второго объекта;
- функция creationFunction для создания и заполнение массива .

Класс MeineKlasse:

- функционал:
    - метод PMassGet — возврат значения поля Array;
    - метод PMassSet — присвоение полю Array значения аргумента.

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 PMassGet класса MeineKlasse

Функционал: возврат указателя на поле Array.

Параметры: нет.

Возвращаемое значение: int* - указатель на поле Array.

Алгоритм метода представлен в таблице 1.

Таблица 1 – Алгоритм метода PMassGet класса MeineKlasse

№	Предикат	Действия	№ перехода
1		Возврат поля Array	∅

3.2 Алгоритм метода PMassSet класса MeineKlasse

Функционал: присвоение полю Array значения аргумента.

Параметры: int* указатель на ячейку памяти.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 2.

Таблица 2 – Алгоритм метода PMassSet класса MeineKlasse

№	Предикат	Действия	№ перехода
1		присвоение значение ptr полю Array	∅

3.3 Алгоритм функции main

Функционал: выполнение поставленной задачи.

Параметры: нет.

Возвращаемое значение: int - код успешности выполнения программы.

Алгоритм функции представлен в таблице 3.

Таблица 3 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		инициализация целочисленной переменной size	2
2		ввод значения size	3
3	size > 2 и size чётный	вывод size и перезод на новую строку	4
		вывод "size?"	19
4		объявление obj1	5
5		значение obj1 = результату выполнения функции creationFunction	6
6		применение метода PairSum для obj1	7
7		инициализация obj2 через конструктор копии	8
8		применение метода PairMult для obj2	9
9		применение метода ArrayOutput для obj1	10
10		вывод результата метода ArraySum для obj1 и переход на новую строчку	11
11		применение метода ArrayOutput для obj2	12
12		вывод результата метода ArraySum для obj2 и переход на новую строчку	13
13		инициализация temp_obj и присвоение ему значения метода PMassGet для obj2	14
14		присвоение obj2 адресс obj1	15
15		приминение метода PMassSet с параметром temp_obj для obj2	16

№	Предикат	Действия	№ перехода
16		приминение метода PairMult для obj1	17
17		вывод результата метода ArraySum для obj2 и переход на новую строчку	18
18		приминение деструктора для obj1, obj2, temp_obj	19
19		возврат 0	∅

3.4 Алгоритм функции creationFunction

Функционал: создаёт и заполняет массив числами.

Параметры: нет.

Возвращаемое значение: int - размер создаваемого массива.

Алгоритм функции представлен в таблице 4.

Таблица 4 – Алгоритм функции creationFunction

№	Предикат	Действия	№ перехода
1		инициализация целочисленного массива по указателю obj	2
2		использование метода ArrayCreate для obj	3
3		использование метода ArrayInput для obj	4
4		использование метода PairMult для obj	5
5		возврат obj	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

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

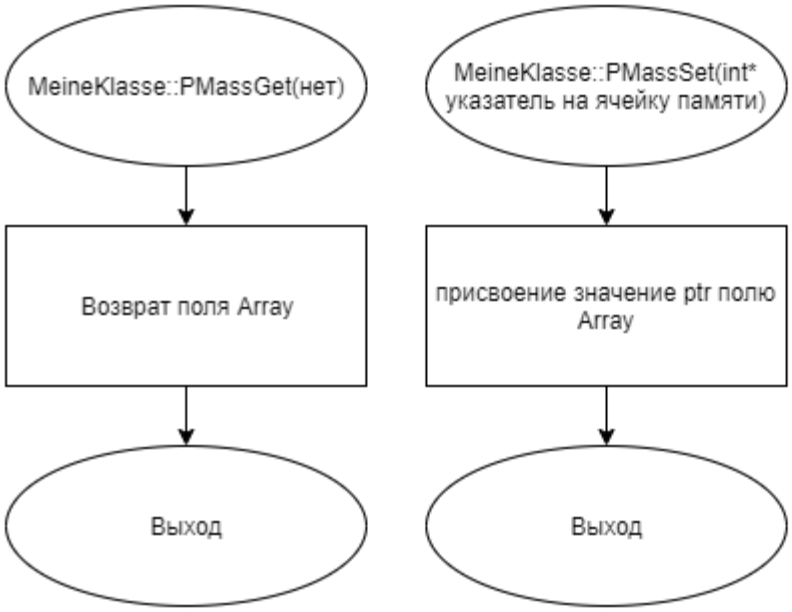

Рисунок 1 – Блок-схема алгоритма

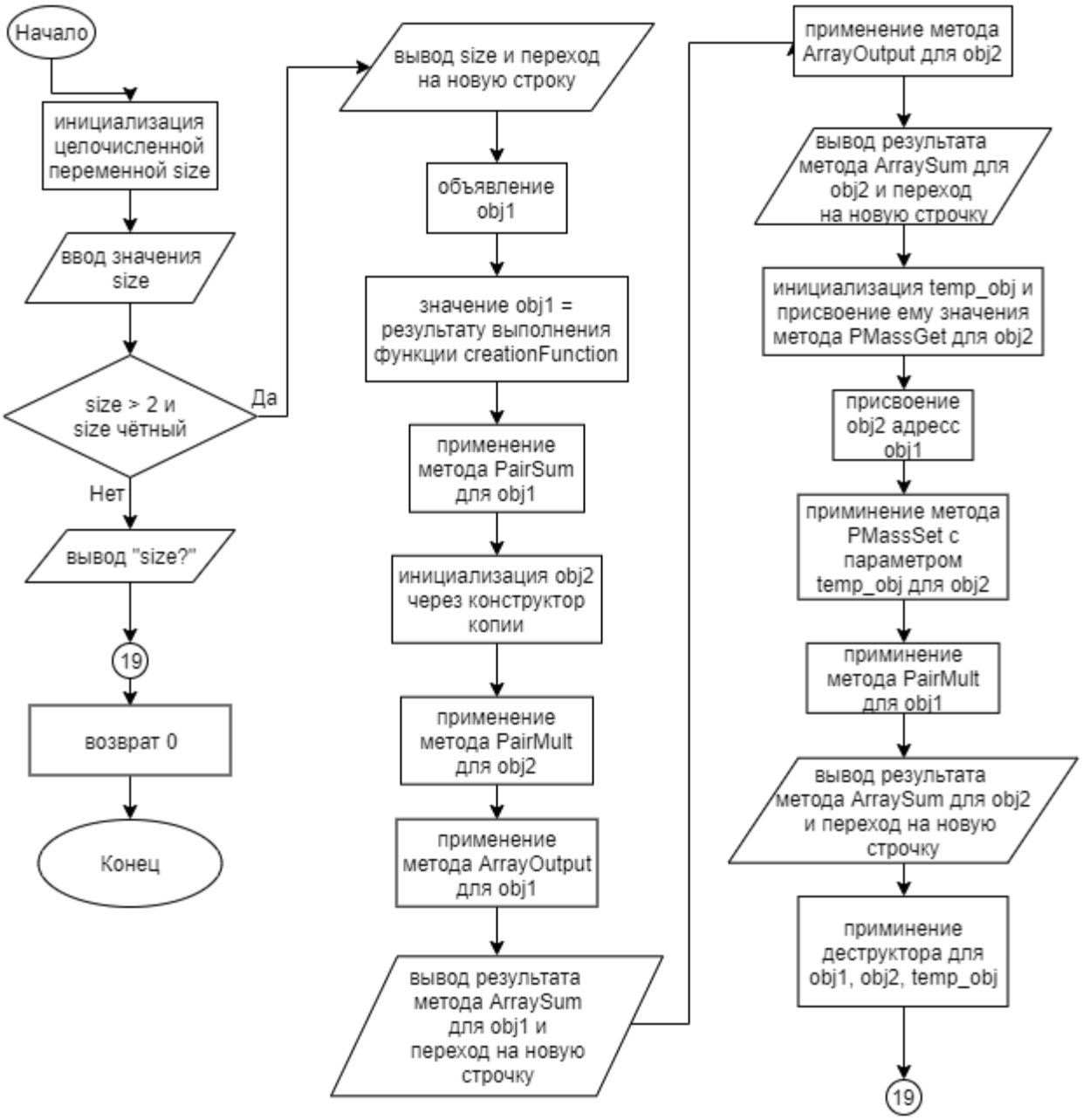

Рисунок 2 – Блок-схема алгоритма

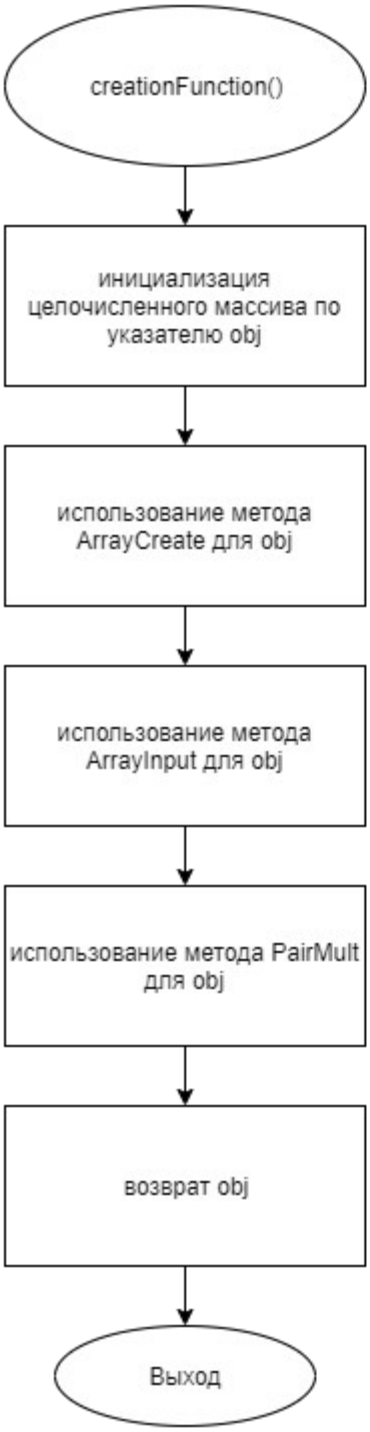

Рисунок 3 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл main.cpp

Листинг 1 – main.cpp

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

MeineKlasse* creationFunction(int size){
   MeineKlasse* obj = new MeineKlasse(size);
   obj->ArrayCreate();
   obj->ArrayInput();
   obj->PairMult();
   return obj;
}

int main()
{
   int size;
   std::cin >> size;
   if ((size > 2) && (size % 2 == 0)){
      std::cout << size << "\n";
      MeineKlasse* obj1;
      obj1 = creationFunction(size);
      obj1->PairSumm();

      MeineKlasse* obj2 = new MeineKlasse(*obj1);
      obj2->PairMult();
      obj1->ArrayOutput();
      std::cout << obj1->ArraySumm() << "\n";
      obj2->ArrayOutput();
      std::cout << obj2->ArraySumm() << "\n";

      int* temp_obj = obj2->PMassGet();
      *obj2 = *obj1;
      obj2->PMassSet(temp_obj);
      obj1->PairMult();
      obj2->ArrayOutput();
      std::cout << obj2->ArraySumm() << "\n";

      delete obj1,
      delete obj2, temp_obj;
```

```
   }
   else{
      std::cout << size << "?\n";
   }
   return 0;
}
```

5.2 Файл MeineKlasse.cpp

Листинг 2 – MeineKlasse.cpp

```
#include "MeineKlasse.h"
#include <iostream>

MeineKlasse::MeineKlasse(){
   Array = nullptr;
   std::cout << "Default constructor\n";
}
MeineKlasse::MeineKlasse(const MeineKlasse& obj){
   Array = new int[obj.size];
   size = obj.size;
   for (int i = 0; i < size; ++i){
      Array[i] = obj.Array[i];
   }
   std::cout << "Copy constructor\n";
}
MeineKlasse::MeineKlasse(int comes_size){
   std::cout << "Constructor set\n";
   this->size = comes_size;
   this->Array = new int[comes_size];
}
MeineKlasse::~MeineKlasse(){
   delete[] Array;
   std::cout << "Destructor\n";
}
void MeineKlasse::ArrayInput(){
   for (int i = 0; i < size; i++){
      std::cin >> Array[i];
   }
}
void MeineKlasse::PairSumm(){
   for (int i = 1; i < size; i+=2){
      Array[i-1] = Array[i] + Array[i-1];
   }
}
void MeineKlasse::PairMult(){
   for (int i = 1; i < size; i+=2){
      Array[i-1] = Array[i] * Array[i-1];
   }
}
```

```
int MeineKlasse::ArraySumm(){
   int summ = 0;
   for (int i = 0; i < size; i++){
      summ += Array[i];
   }
   return summ;
}
void MeineKlasse::ArrayCreate(){
   Array = new int[size];
}
void MeineKlasse::ArrayOutput(){
   for (int i = 0; i < size - 1; ++i){
      std::cout << Array[i] << "  ";
   }
   std::cout << Array[size-1] << "\n";
}
int* MeineKlasse::PMassGet(){
   return Array;
}
void MeineKlasse::PMassSet(int* ptr){
   Array = ptr;
}
```

5.3 Файл MeineKlasse.h

Листинг 3 – MeineKlasse.h

```
#ifndef __MEINEKLASSE__H
#define __MEINEKLASSE__H

class MeineKlasse{
public:
   MeineKlasse();
   MeineKlasse(const MeineKlasse& obj);
   MeineKlasse(int size);
   ~MeineKlasse();
   void ArrayInput();
   void PairSumm();
   void PairMult();
   int ArraySumm();
   void ArrayOutput();
   void ArrayCreate();

   int* PMassGet();
   void PMassSet(int* ptr);
private:
   int size;
   int* Array;
};
```

```
    #endif
```

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. 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).