|  |  |
| --- | --- |
| МИРЭА_ЭМБЛЕМА_приказ | |
| МИНОБРАНАУКИ РОССИИ | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  «**МОСКОВСКИЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»**  **МИРЭА** | |
| Институт информационных технологий  Кафедра математического обеспечения и стандартизации информационных технологий | |
| **Отчет по лабораторной работе №1**  **Тема работы «Одномерный массив»** | |
| Выполнил: |  |
| Студент группы ИКБО-12-17  Отчет принял:  Преподаватель кафедры МОСИТ | Лисовой А.А.  Миронов Антон Николаевич |
| Москва 2018 | |

**Оглавление**

[Задание 1 2](#_Toc507327653)

[Декомпозиция 2](#_Toc507327654)

[Определение функций: 2](#_Toc507327655)

[Реализация функций 3](#_Toc507327656)

[Задание 2 3](#_Toc507327657)

[Декомпозиция 4](#_Toc507327658)

[Определение функций: 5](#_Toc507327659)

[Реализация функций 7](#_Toc507327660)

[Кодирование алгоритма программы 9](#_Toc507327661)

# Задание 1

Условие задания :

Дан одномерный целочисленный массив А состоящий из N элементов(0<N<20;-10000<A[i]<10000).. найти наибольший среди элементов массива А, являющихся простыми числами.

# Декомпозиция

Задачу следует разбить на следующие подзадачи:

* Проверка элемента массива на принадлежность к простым числам;
* Поиск наибольшего элемента массива;

# Определение функций:

1. Проверка элемента массива на принадлежность к простым числам:

bool Task1\_PrimeNumber(int N);

A

N – int

Логическое значение

Входные данные

Результат

Task1\_PrimeNumber

1. Нахождение наибольшего элемента массива:

void Task1\_MaxPrimeNumber(int\*Array, int Length);

A

Array – int\*

Length - int

Max - int

Входные данные

Результат

Task1\_MaxPrimeNumber

# Реализация функций

bool Task1\_PrimeNumber(int N)

{

for (int i = 2; i <= 10; i++)

{

bool check = false;

if (i == N) continue;

if (N%i != 0) check = true;

if (check == false || N == 1 || N == 0)

{

return false;

}

}

return true;

}

void Task1\_MaxPrimeNumber(int\*Array, int Length)

{

if (Length == NULL)

{

cout << "Массив не был введен!\n";

cout << "Введите массив и попробуйте ещё раз.\n";

system("pause");

}

else

{

int Max = 0;

for (int i = 0; i<Length; i++)

{

if (Task1\_PrimeNumber(Array[i]) == true && Array[i] > Max)

Max = Array[i];

}

cout << "Наибольший среди элементов массива, являющийся простым числом: " << Max << endl;

system("pause");

}

}

# Задание 2

Условие задания:

Дан массив A, состоящий из N целых чисел

* + Сформировать массив В из неповторяющихся элементов массива А.
  + Удалить из массива А элементы, шестнадцатеричный код которых содержит буквы.
  + Упорядочить массив по правилу: с младших индексов числа начинающиеся с цифры 1, затем 2, затем 3 и т.д.( сохраняя взаимное расположение всех перемещаемых элементов).
  + Найти НОД всех элементов массива, сумма цифр которых кратна 7.

# Декомпозиция

Задачу следует разбить на следующие подзадачи:

* Сформировать и вывести массив В, содержащий неповторяющиеся элементы массива А.
* Удалить из массива А элементы, шестнадцатеричный код которых содержит буквы.
* Упорядочить массив по правилу: с младших индексов числа начинающиеся с цифры 1, затем 2, затем 3 и т.д:

1. Определить старший разряд элемента массива;
2. Упорядочить массив по указанному правилу;

* Найти НОД всех элементов массива, сумма цифр которых кратна 7:

1. Определить, кратна ли сумма цифр элемента массива 7;
2. Записать подходящие элементы в новый массив;
3. Найти НОД массива элементов, сумма цифр которых кратна 7;

# 

# Определение функций:

1. Сформировать и вывести массив В, содержащий неповторяющиеся элементы массива А.

void Task2\_RemoveRepeate(int\*Array, int Length)

A

Array – int\*

Length – int;

DynamicArrayB - int

Входные данные

Результат

Task2\_RemoveRepeate

1. Удалить из массива А элементы, шестнадцатеричный код которых содержит буквы.

int Task2\_RemoveSymbolsInUnicode(int\*Array, int Length)

A

Array – int\*

Length – int;

DynamicArray - int

Length – int;

Входные данные

Результат

Task2\_RemoveSymbolsInUnicode

1. Упорядочить массив по правилу: с младших индексов числа начинающиеся с цифры 1, затем 2, затем 3 и т.д:
2. Определить старший разряд элемента массива:

int Task2\_MostSignificantBit(int Number)

A

Number – int

Number - int

Входные данные

Результат

Task2\_MostSignificantBit

1. Упорядочить массив по указанному правилу:

void Task2\_BubbleIndex(int\*Array, int Length)

A

Array – int\*

Length – int;

DynamicArray - int

Входные данные

Результат

Task2\_BubbleIndex

1. Найти НОД всех элементов массива, сумма цифр которых кратна 7:
2. Определить, кратна ли сумма цифр элемента массива:

bool Task2\_SumOfNumbersBySeven(int Number)

A

Number – int

Логическое значение

Входные данные

Результат

Task2\_SumOfNumbersBySeven

1. Записать подходящие элементы в новый массив:

void Task2\_SumBySevenArray(int\*Array, int Length)

A

Array – int\*

Length – int;

DynamicArrayB - int

Входные данные

Результат

Task2\_SumBySevenArray

1. Найти среди них НОД:

void Task2\_NOD(int\*Array, int Length)

A

Array – int\*

Length – int;

NOD - int

Входные данные

Результат

Task2\_NOD

# Реализация функций

void Task2\_RemoveRepeate(int\*Array, int Length)

{

system("cls");

cout << "Массив без повторяющихся значнений: " << endl;

bool flag = false;

int SubLength = 0;

int \*DynamicArrayB = new int[Length];

for (int i = 0; i < Length; ++i)

{

flag = false;

for (int j = 0; j < Length; ++j) {

if (i == j)

continue;

else

{

if (Array[i] == Array[j])

{

flag = false;

break;

}

else

flag = true;

}

}

if (flag == true)

DynamicArrayB[SubLength++] = Array[i];

}

for (int i = 0; i < SubLength; ++i)

cout << "DynamicArrayB[" << i << "]" << "= " << DynamicArrayB[i] << endl;

delete DynamicArrayB;

cout << "Вывод массива успешно осуществлен!" << endl;

system("pause");

}

int Task2\_RemoveSymbolsInUnicode(int\*Array, int Length)

{

cout << "Удаляемый диапазон элементов массива:" << endl;

cout << "Для букв латиницы: (0,127)" << endl;

cout << "Для букв кириллицы: (1024,1279)" << endl;

int i,j;

for (i = 0; i < Length; ++i)

if ((Array[i] < 0 || Array[i] < 128) || (Array[i] > 1023 && Array[i] < 1280))

{

for (j = i; j < Length; ++j)

Array[j] = Array[j + 1];

Length--;

i--;

}

Task2\_ArrayOutput(Array, Length);

return Length;

}

int Task2\_MostSignificantBit(int Number)

{

while (Number > 9)

{

Number = Number / 10;

}

return Number;

}

void Task2\_BubbleIndex(int\*Array, int Length)

{

cout << "Отсортированный согласно условию массив:" << endl;

int temp;

int Number, IncNumber;

for (int i = 0; i < Length - 1; i++) {

for (int j = 0; j < Length - i - 1; j++) {

if ((Number = Task2\_MostSignificantBit(Array[j])) > (IncNumber = Task2\_MostSignificantBit(Array[j + 1])))

{

temp = Array[j];

Array[j] = Array[j + 1];

Array[j + 1] = temp;

}

}

}

Task2\_ArrayOutput(Array, Length);

}

bool Task2\_SumOfNumbersBySeven(int Number)

{

int Sum = 0;

while (Number != 0)

{

Sum = Sum + (Number % 10);

Number = Number / 10;

}

if (Sum % 7 == 0)

{

return true;

}

else

{

return false;

}

}

void Task2\_SumBySevenArray(int\*Array, int Length)

{

int SubLength = 0;

int \*DynamicArrayB = new int[Length];

int i;

for (i = 0; i < Length; ++i)

if (Task2\_SumOfNumbersBySeven(Array[i]))

{

DynamicArrayB[SubLength++] = Array[i];

}

Task2\_NOD(DynamicArrayB, SubLength);

delete DynamicArrayB;

}

void Task2\_NOD(int\*Array, int Length)

{

system("cls");

int NOD = 0;

for (int i = 0; i < Length; i++)

{

if (NOD < Array[i])

NOD = Array[i];

}

while (NOD > 1)

{

bool flag = true;

for (int i = 0; i < Length; i++)

if (Array[i] % NOD != 0)

{

flag = false;

break;

}

if (!flag) --NOD;

else

break;

}

if (NOD == 0)

{

cout << "В массиве не оказалось элементов, сумма которых кратна 7!" << endl;

system("pause");

}

else

{

cout << "НОД для элементов массива, сумма цифр которых кратна 7, равен: " << NOD << endl;

system("pause");

}

}

# Кодирование алгоритма программы

#include "stdafx.h"

#include <iostream>

#include <stdio.h>

#include <locale.h>

#include <malloc.h>

using namespace std;

#define MaxLength 20

int Array[MaxLength];

int Length;

//ПРОТОТИПЫ ФУНКЦИЙ

int Interface\_Main();

int Interface\_Task1();

int Interface\_Task1\_InputArray();

int Interface\_Task2();

int Interface\_Task2\_InputArray();

void Task1\_InputArrayByHand(int\*Array, int Length);

void Task2\_InputArrayByHand(int\*Array, int Length);

void Task1\_InputArrayByRand(int\*Array, int Length);

void Task2\_InputArrayByRand(int\*Array, int Length);

void Task1\_ArrayOutput(int\*Array, int Length);

void Task2\_ArrayOutput(int\*Array, int Length);

bool Task1\_PrimeNumber(int N);

void Task1\_MaxPrimeNumber(int\*Array, int Length);

void Task2\_RemoveRepeate(int\*Array, int Length);

int Task2\_RemoveSymbolsInUnicode(int\*Array, int Length);

int Task2\_MostSignificantBit(int Number);

void Task2\_BubbleIndex(int\*Array, int Length);

bool Task2\_SumOfNumbersBySeven(int Number);

void Task2\_SumBySevenArray(int\*Array, int Length);

void Task2\_NOD(int\*Array, int Length);

int main()

{

system("cls");

setlocale(LC\_CTYPE, "RUS");

int\* DynamicArray = (int\*)malloc(Length \* sizeof(int));

int step;

int back\_main = 0;

int back\_sub = 0;

while (back\_main == 0)

{

step = Interface\_Main();

switch (step)

{

case 1:

system("cls");

cout << "Выбрано задание 1" << endl;

while (back\_sub == 0)

{

step = Interface\_Task1();

switch (step)

{

/\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*/

case 1:

system("cls");

cout << "Введите длину массива:" << endl;

cout << ">>> ";

cin >> Length;

if (Length > 0 && Length <= MaxLength)

{

while (back\_sub == 0)

{

step = Interface\_Task1\_InputArray();

switch (step)

{

case 1:

system("cls");

Task1\_InputArrayByHand(Array, Length);

back\_sub = 1;

break;

case 2:

system("cls");

Task1\_InputArrayByRand(Array, Length);

back\_sub = 1;

break;

default:

back\_sub = 1;

break;

}

}

back\_sub = 0;

}

else

{

system("cls");

cout << "Длина должна массива быть больше 0 и меньше 20!" << endl;

cout << "Попробуйте еще раз." << endl;

system("pause");

}

break;

case 2:

system("cls");

if (Length == NULL)

{

cout << "Массив не был введен!\n";

cout << "Введите массив и попробуйте ещё раз.\n";

system("pause");

break;

}

else

{

Task1\_ArrayOutput(Array, Length);

break;

}

case 3:

system("cls");

Task1\_MaxPrimeNumber(Array, Length);

break;

case 4:

Length = NULL;

back\_sub = 1;

break;

default:

back\_sub = 1;

break;

}

}

back\_sub = 0;

break;

/\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*/

case 2:

system("cls");

cout << "Выбрано задание 2" << endl;

while (back\_sub == 0)

{

step = Interface\_Task2();

switch (step)

{

case 1:

system("cls");

cout << "Введите длину массива:" << endl;

cout << ">>> ";

cin >> Length;

if (Length > 0)

{

DynamicArray = (int\*)realloc(DynamicArray, sizeof(int)\*(Length));

while (back\_sub == 0)

{

step = Interface\_Task2\_InputArray();

switch (step)

{

case 1:

system("cls");

Task2\_InputArrayByHand(DynamicArray, Length);

back\_sub = 1;

break;

case 2:

system("cls");

Task2\_InputArrayByRand(DynamicArray, Length);

back\_sub = 1;

break;

default:

back\_sub = 1;

break;

}

}

back\_sub = 0;

}

else

{

system("cls");

cout << "Длина должна массива быть больше 0!" << endl;

cout << "Попробуйте еще раз." << endl;

system("pause");

}

break;

case 2:

system("cls");

if (Length == NULL)

{

cout << "Массив не был введен!\n";

cout << "Введите массив и попробуйте ещё раз.\n";

system("pause");

break;

}

else

{

Task2\_ArrayOutput(DynamicArray, Length);

break;

}

case 3:

system("cls");

if (Length == NULL)

{

cout << "Массив не был введен!\n";

cout << "Введите массив и попробуйте ещё раз.\n";

system("pause");

break;

}

else

{

Task2\_RemoveRepeate(DynamicArray, Length);

break;

}

case 4:

system("cls");

if (Length == NULL)

{

cout << "Массив не был введен!\n";

cout << "Введите массив и попробуйте ещё раз.\n";

system("pause");

break;

}

else

{

Length = Task2\_RemoveSymbolsInUnicode(DynamicArray, Length);

break;

}

case 5:

system("cls");

if (Length == NULL)

{

cout << "Массив не был введен!\n";

cout << "Введите массив и попробуйте ещё раз.\n";

system("pause");

break;

}

else

{

Task2\_BubbleIndex(DynamicArray, Length);

break;

}

case 6:

system("cls");

if (Length == NULL)

{

cout << "Массив не был введен!\n";

cout << "Введите массив и попробуйте ещё раз.\n";

system("pause");

break;

}

else

{

Task2\_SumBySevenArray(DynamicArray, Length);

break;

}

case 7:

back\_sub = 1;

Length = NULL;

break;

default:

back\_sub = 1;

break;

}

}

back\_sub = 0;

break;

/\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*/

case 3:

system("cls");

system("pause");

back\_main = 1;

break;

/\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*/

default:

system("cls");

cout << "Ошибка ввода!" << endl;

cout << "Работа программы остановлена." << endl;

system("pause");

back\_main = 1;

break;

/\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*/

}

}

delete DynamicArray;

}

//ОПИСАНИЕ ФУНКЦИЙ ИНТЕРФЕЙСА

int Interface\_Main()

{

system("cls");

int step;

cout << "Лабораторная работа №1: Вариант 14" << endl;

cout << "Выберите действие:" << endl;

cout << "1. Задание 1;\n";

cout << "2. Задание 2;\n";

cout << "3. Выход из программы.\n";

cout << ">>> ";

cin >> step;

return step;

}

int Interface\_Task1()

{

system("cls");

int step;

cout << "Выберите действие:" << endl;

cout << "1. Ввод данных в массив.\n";

cout << "2. Вывод массива.\n";

cout << "3. Найти наибольший среди элементов массива, являющихся простыми числами.\n";

cout << "4. Вернуться к выбору задания.\n";

cout << ">>> ";

cin >> step;

return step;

}

int Interface\_Task1\_InputArray()

{

int step;

cout << "Выберите действие:" << endl;

cout << "1. Ввод данных в массив вручную.\n";

cout << "2. Заполнение массива случайными числами.\n";

cout << ">>> ";

cin >> step;

return step;

}

int Interface\_Task2()

{

system("cls");

int step;

cout << "Выберите действие:" << endl;

cout << "1. Ввод данных в массив.\n";

cout << "2. Вывод массива.\n";

cout << "3. Сформировать массив В из неповторяющихся элементов массива А.\n";

cout << "4. Удалить из массива А элементы, шестнадцатеричный код которых содержит буквы.\n";

cout << "5. Упорядочить массив по правилу: с младших индексов числа начинающиеся с цифры 1, затем 2, затем 3 и т.д.\n";

cout << "6. Найти НОД всех элементов массива, сумма цифр которых кратна 7.\n";

cout << "7. Вернуться к выбору задания.\n";

cout << ">>> ";

cin >> step;

return step;

}

int Interface\_Task2\_InputArray()

{

int step;

cout << "Выберите действие:" << endl;

cout << "1. Ввод данных в массив вручную.\n";

cout << "2. Заполнение массива случайными числами.\n";

cout << ">>> ";

cin >> step;

return step;

}

//ОПИСАНИЕ ФУНКЦИЙ

void Task1\_InputArrayByHand(int\*Array, int Length)

{

bool mistake = false;

cout << "Введите данные в массив:" << endl;

for (int i = 0; i < Length; i++)

{

cin >> Array[i];

if (Array[i] < -10000 || Array[i] > 10000)

{

mistake = true;

break;

}

}

if (mistake == false)

{

cout << "Ввод окончен!" << endl;

system("pause");

}

else

{

system("cls");

cout << "Выход значения элемента массива за допустимый диапазон!" << endl;

cout << "Заполните массив заново." << endl;

system("pause");

}

}

void Task2\_InputArrayByHand(int\*Array, int Length)

{

bool mistake = false;

cout << "Введите данные в массив:" << endl;

for (int i = 0; i < Length; i++)

{

cin >> Array[i];

if (Array[i] < 0 || Array[i] > 2000)

{

mistake = true;

break;

}

}

if (mistake == false)

{

cout << "Ввод окончен!" << endl;

system("pause");

}

else

{

system("cls");

cout << "Выход значения элемента массива за допустимый диапазон(0...2000)!" << endl;

cout << "Заполните массив заново." << endl;

system("pause");

}

}

void Task1\_InputArrayByRand(int\*Array, int Length)

{

for (int i = 0; i < Length; i++)

Array[i] = -10001 + rand() % 20000;

cout << "Массив заполнен случайными числами в допустимом диапазоне!" << endl;

system("pause");

}

void Task2\_InputArrayByRand(int\*Array, int Length)

{

for (int i = 0; i < Length; i++)

Array[i] = rand() % 2000;

cout << "Массив заполнен случайными числами в допустимом диапазоне(0...2000)!" << endl;

system("pause");

}

void Task1\_ArrayOutput(int\*Array, int Length)

{

cout << "Массив:" << endl;

for (int i = 0; i < Length; i++)

cout << "Array[" << i << "]" << "= " << Array[i] << endl;

cout << endl;

cout << "Вывод массива успешно осуществлен!" << endl;

system("pause");

}

void Task2\_ArrayOutput(int\*Array, int Length)

{

cout << "Массив:" << endl;

for (int i = 0; i < Length; i++)

cout << "DynamicArray[" << i << "]" << "= " << Array[i] << endl;

cout << endl;

cout << "Вывод массива успешно осуществлен!" << endl;

system("pause");

}

bool Task1\_PrimeNumber(int N)

{

for (int i = 2; i <= 10; i++)

{

bool check = false;

if (i == N) continue;

if (N%i != 0) check = true;

if (check == false || N == 1 || N == 0)

{

return false;

}

}

return true;

}

void Task1\_MaxPrimeNumber(int\*Array, int Length)

{

if (Length == NULL)

{

cout << "Массив не был введен!\n";

cout << "Введите массив и попробуйте ещё раз.\n";

system("pause");

}

else

{

int Max = 0;

for (int i = 0; i<Length; i++)

{

if (Task1\_PrimeNumber(Array[i]) == true && Array[i] > Max)

Max = Array[i];

}

cout << "Наибольший среди элементов массива, являющийся простым числом: " << Max << endl;

system("pause");

}

}

void Task2\_RemoveRepeate(int\*Array, int Length)

{

system("cls");

cout << "Массив без повторяющихся значнений: " << endl;

bool flag = false;

int SubLength = 0;

int \*DynamicArrayB = new int[Length];

for (int i = 0; i < Length; ++i)

{

flag = false;

for (int j = 0; j < Length; ++j) {

if (i == j)

continue;

else

{

if (Array[i] == Array[j])

{

flag = false;

break;

}

else

flag = true;

}

}

if (flag == true)

DynamicArrayB[SubLength++] = Array[i];

}

for (int i = 0; i < SubLength; ++i)

cout << "DynamicArrayB[" << i << "]" << "= " << DynamicArrayB[i] << endl;

delete DynamicArrayB;

cout << "Вывод массива успешно осуществлен!" << endl;

system("pause");

}

int Task2\_RemoveSymbolsInUnicode(int\*Array, int Length)

{

cout << "Удаляемый диапазон элементов массива:" << endl;

cout << "Для букв латиницы: (0,127)" << endl;

cout << "Для букв кириллицы: (1024,1279)" << endl;

int i,j;

for (i = 0; i < Length; ++i)

if ((Array[i] < 0 || Array[i] < 128) || (Array[i] > 1023 && Array[i] < 1280))

{

for (j = i; j < Length; ++j)

Array[j] = Array[j + 1];

Length--;

i--;

}

Task2\_ArrayOutput(Array, Length);

return Length;

}

int Task2\_MostSignificantBit(int Number)

{

while (Number > 9)

{

Number = Number / 10;

}

return Number;

}

void Task2\_BubbleIndex(int\*Array, int Length)

{

cout << "Отсортированный согласно условию массив:" << endl;

int temp;

int Number, IncNumber;

for (int i = 0; i < Length - 1; i++) {

for (int j = 0; j < Length - i - 1; j++) {

if ((Number = Task2\_MostSignificantBit(Array[j])) > (IncNumber = Task2\_MostSignificantBit(Array[j + 1])))

{

temp = Array[j];

Array[j] = Array[j + 1];

Array[j + 1] = temp;

}

}

}

Task2\_ArrayOutput(Array, Length);

}

bool Task2\_SumOfNumbersBySeven(int Number)

{

int Sum = 0;

while (Number != 0)

{

Sum = Sum + (Number % 10);

Number = Number / 10;

}

if (Sum % 7 == 0)

{

return true;

}

else

{

return false;

}

}

void Task2\_SumBySevenArray(int\*Array, int Length)

{

int SubLength = 0;

int \*DynamicArrayB = new int[Length];

int i;

for (i = 0; i < Length; ++i)

if (Task2\_SumOfNumbersBySeven(Array[i]))

{

DynamicArrayB[SubLength++] = Array[i];

}

Task2\_NOD(DynamicArrayB, SubLength);

delete DynamicArrayB;

}

void Task2\_NOD(int\*Array, int Length)

{

system("cls");

int NOD = 0;

for (int i = 0; i < Length; i++)

{

if (NOD < Array[i])

NOD = Array[i];

}

while (NOD > 1)

{

bool flag = true;

for (int i = 0; i < Length; i++)

if (Array[i] % NOD != 0)

{

flag = false;

break;

}

if (!flag) --NOD;

else

break;

}

if (NOD == 0)

{

cout << "В массиве не оказалось элементов, сумма которых кратна 7!" << endl;

system("pause");

}

else

{

cout << "НОД для элементов массива, сумма цифр которых кратна 7, равен: " << NOD << endl;

system("pause");

}

}