**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное образовательное учреждение высшего образования**

**МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕХНОЛОГИЙ И УПРАВЛЕНИЯ**

**УНИВЕРСИТЕТСКИЙ КОЛЛЕДЖ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

**ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ №3**

«Одномерные массивы»

Студента 2 курса, ИСП-208

Каршибаева Жахонгира

Направление 09.02.07 – «Информационные системы и программирование»

Руководитель:

Преподаватель

М.В. Валеев

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Работа защищена

«\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»

“\_\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_2022г.

Москва 2022

Оглавление

[1.Номер первый 5](#_Toc103410985)

[1.1. Постановка задачи 5](#_Toc103410986)

[1.2. Структурные требование 5](#_Toc103410987)

[1.3. Описание работы 5](#_Toc103410988)

[1.4.1. Использование переменных 5](#_Toc103410989)

[1.4.2. Псевдокод программы 5](#_Toc103410990)

[1.4.3 Блок-схема алгоритма 6](#_Toc103410991)

[1.4.4. Листинг программной реализации 7](#_Toc103410992)

[1.4.5. Результаты 8](#_Toc103410993)

[1.4.6. Тесты 8](#_Toc103410994)

[2.Номер второй 9](#_Toc103410995)

[Постановка задачи 9](#_Toc103410996)

[2.2. Структурные требование 9](#_Toc103410997)

[2.3. Описание работы 9](#_Toc103410998)

[2.4.1. Использование переменных 9](#_Toc103410999)

[2.4.2. Псевдокод программы 9](#_Toc103411000)

[2.4.3. Блок-схема алгоритма 10](#_Toc103411001)

[2.4.4. Листинг программной реализации 11](#_Toc103411002)

[2.4.5. Результаты 12](#_Toc103411003)

[2.4.6. Тесты 13](#_Toc103411004)

[3.Номер третий 14](#_Toc103411005)

[3.1. Постановка задачи 14](#_Toc103411006)

[3.2. Структурные требование 14](#_Toc103411007)

[3.3. Описание работы 14](#_Toc103411008)

[3.4.1. Использование переменных 14](#_Toc103411009)

[3.4.2. Псевдокод программы 14](#_Toc103411010)

[3.4.3. Блок-схема алгоритма 14](#_Toc103411011)

[3.4.4. Листинг программной реализации 16](#_Toc103411012)

[3.4.5. Результаты 17](#_Toc103411013)

[3.4.6. Тесты 18](#_Toc103411014)

[4.Номер четвертый 19](#_Toc103411015)

[4.1. Постановка задачи 19](#_Toc103411016)

[4.2. Структурные требование 19](#_Toc103411017)

[3.3. Описание работы 19](#_Toc103411018)

[4.4.1. Использование переменных 19](#_Toc103411019)

[4.4.2. Псевдокод программы 19](#_Toc103411020)

[4.4.3. Блок-схема алгоритма 19](#_Toc103411021)

[4.4.4. Листинг программной реализации 21](#_Toc103411022)

[4.4.5. Результаты 21](#_Toc103411023)

[4.4.6. Тест 22](#_Toc103411024)

[5.Номер пяты 23](#_Toc103411025)

[5.1. Постановка задачи 23](#_Toc103411026)

[5.2. Структурные требование 23](#_Toc103411027)

[5.3. Описание работы 23](#_Toc103411028)

[5.4.1. Использование переменных 23](#_Toc103411029)

[5.4.2. Псевдокод программы 23](#_Toc103411030)

[5.4.3. Блок-схема алгоритма 24](#_Toc103411031)

[5.4.4. Листинг программной реализации 26](#_Toc103411032)

[5.4.5. Результаты 27](#_Toc103411033)

[5.4.6. Тесты 27](#_Toc103411034)

[Список литературы 28](#_Toc103411035)

1. **Постановка задачи №1**

Найти максимальный и минимальный элементы массива

1. **Структурные требовани**

Max\_n>0. Задается как константа

1. **Описание работы**
   1. Используемые переменные

i - переменная для цикла

min - минимальное значение массива

max - максимальное значение

* 1. Псевдокод программы

int a[MAX\_N] константа

for (i=0; i<10; i++) заполняем массив через цикл for до MAX\_N, а затем присваиваем первой переменной значение мин и макс min =a[0];max=a[0]. Затем пробегаем по массиву и имщем минималное значение, если находим то присваиваем новое значени мин if (a[i]<min){min=a[i];}; точно так же и с макс. И выводим значения.

* 1. Блок - схема алгоритма

Введите элементы массива

i=0; I<MAX\_N; I++

MAX\_N

Начало

Конец

max; min

max=a[i];

a[i]>max

min=a[i];

a[i]<min

i=1; i<MAX\_N; i++

min = a[0]

max=a[0]

* 1. Листинг программной реализации

#include<stdio.h>

#include<conio.h>

int main()

{

int MAX\_N;

scanf("%d", &MAX\_N);

int a[MAX\_N];// объявление целочисленного массива a из 10 элементов

int i, min, max;

// ввод элементов массива с клавиатуры

for (i=0; i<MAX\_N; i++)

{

printf("Введите элемент mассива %d\n", i);

scanf("%d", &a[i]);

}

min =a[0];// присваиваем минимальному значению значение первого элемента

max=a[0];// присваиваем максимальному значению значение первого элемента

// пробегаем по всем элементам массива

for (i=1; i<MAX\_N; i++)

{

// если текущий элемент массива меньше текущего минимума то присваиваем его значение минимуму

if (a[i]<min)

{

min=a[i];

}

// если текущий элемент массива больше текущего максимума то присваиваем его значение максимуму

if (a[i]>max)

{

max=a[i];

}

}

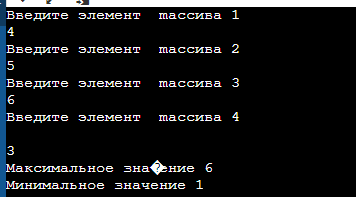
// выводим значения на экран

printf("Максимальное значение %d\n",max);

printf("Минимальное значение %d\n",min);

return 0; }

* 1. Результат



* 1. Тесты

MAX\_N = 5

|  |  |
| --- | --- |
| 1 = 5 | max = 78 |
| 2 = 4 |
| 3 = 78 |
| 4 = 9 | min = 4 |
| 5 = 64 |

|  |  |
| --- | --- |
| 1 = 1 | max = 79 |
| 2 = 4 |
| 3 = 8 |
| 4 = 79 | min = 1 |
| 5 = 64 |

1. **Постановка задачи №2**

Переставить местами максимальный и минимальный элементы

1. **Структурные требования**

Максимальная размерность массива MAX\_N задается глобальной

константой

Текущая размерность массива n вводится в программе через оператор

Ввода

MAX\_n >0

1. **Описание работы** 
   1. Использованные переменные

i, max, min, P

* 1. Псевдокод программы

Начало. Вводим MAX\_N как константу и вводим его, будет количеством. После чего мы заполняем массив через клавиатуру и начинаем проверку массива. присваиваем первой переменной значение мин и макс min =a[0];max=a[0]. Затем пробегаем по массиву и имщем минималное значение, если находим то присваиваем новое значени мин if (a[i]<min){min=a[i];}; точно так же и с макс. И выводим значения. После чего присваиваем мин значени Р, а макс значение мин. Выводим значени макс и мин. Конец

* 1. Блок схема алгоритма

i=0; I<MAX\_N; I++

MAX\_N

Начало

i=1; i<MAX\_N; i++

min = a[0]

max=a[0]

Введите элементы массива

max=a[i];

a[i]>max

min=a[i];

a[i]<min

P=min;

Min = max

Max = p

Конец

max; min

* 1. Листинг программной реализации

#include<stdio.h>

#include<conio.h>

int main()

{

int MAX\_N;

scanf("%d", &MAX\_N);

int a[MAX\_N];// объявление целочисленного массива a из 10 элементов

int i, min, max, P;

// ввод элементов массива с клавиатуры

for (i=0; i<MAX\_N; i++)

{

printf("Введите элемент mассива %d\n", i);

scanf("%d", &a[i]);

}

min =a[0];// присваиваем минимальному значению значение первого элемента

max=a[0];// присваиваем максимальному значению значение первого элемента

// пробегаем по всем элементам массива

for (i=1; i<MAX\_N; i++)

{

// если текущий элемент массива меньше текущего минимума то присваиваем его значение минимуму

if (a[i]<min){min=a[i];};

// если текущий элемент массива больше текущего максимума то присваиваем его значение максимуму

if (a[i]>max){max=a[i];};

}

P=min; //значение мин сохраняем в Р потом нам это понадобится

min = max; //значению мин присваеваем значени макс

max = P; //значение макс присваеваем Р, точнее мин

// выводим значения на экран

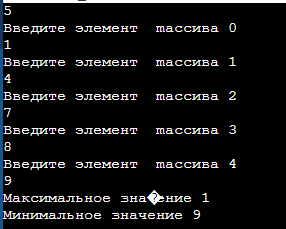
printf("Максимальное значение %d\n",max);

printf("Минимальное значение %d\n",min);

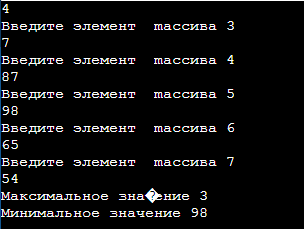
return 0;

}

* 1. Результаты



* 1. Тесты



1. **Постановка задачи №3**

Найти количество различных элементов в массиве. Вывести значения неповторяющихся элементов.

1. **Структурные требования**

Максимальная размерность массива MAX\_N задается глобальной

константой

Текущая размерность массива n вводится в программе через оператор Ввода n>0

1. **Описание работы**
   1. Используемые переменные

M, a, c, n, I, j

* 1. Псевдокод программы

Начало. Объявляем переменные и заполняем массив рандом M[i] = ((RAND\_MAX) / rand()). Затем пробегаем по массиву в поисках различных. Проверяем массив I на наличие различных элементов. Затем выводим сам массив и количество различных.

* 1. Блок схема

Начало

Введите кол-во данных

n<0

Введите число больше 0

i = 0; i < n;

M[i] = ((RAND\_MAX) / rand())

M[i]

i = 0; i < n;

a = 1

j = 0; j < n; j++

M[i] == M[j]) && (i != j

а=0

a == 1

M[i]

C++

Кол-во различных элемент ов в массиве

Конец

* 1. Листнг программной реализации

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

int main()

{

srand(time(NULL));

int\* M, a, c = 0, n;

printf(" Введите кол-во элементов: \n");

scanf("%i", &n);

if (n < 1) printf("Введите число больше 0\n");

else

{

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

{

M[i] = ((RAND\_MAX) / rand());

printf(" %i", M[i]);

}

printf("\n Неповторяющиеся элементы: ");

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

{

a = 1;

for (int j = 0; j < n; j++)

{

if ((M[i] == M[j]) && (i != j))

{

a = 0;

}

}

if (a == 1)

{

printf(" %i", M[i]);

c++;

}

}

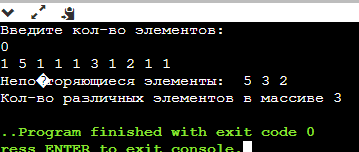
printf("\n Кол-во различных элементов в массиве %i", c);

}

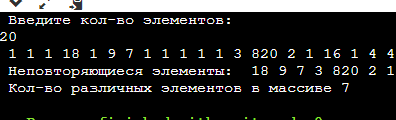
return 0;

}

* 1. Результат



* 1. Тесты



1. **Постановка задачи №4**

В одномерном массиве, состоящем из п целых элементов, вычислить: минимальный по модулю элемент массива;

1. **Структурные требования**

Максимальная размерность массива MAX\_N задается глобальной

константой

Текущая размерность массива n вводится в программе через оператор Ввода n>0

1. **Описание работы** 
   1. Использованные переменные

i, min, max

* 1. Псевдокод программы

int a[MAX\_N] константа

for (i=0; i<10; i++) заполняем массив через цикл for до MAX\_N, а затем присваиваем первой переменной значение мин и макс min =a[0];max=a[0]. Затем пробегаем по массиву и имщем минималное значение, если находим то присваиваем новое значени мин if (a[i]<min){min=a[i];}; точно так же и с макс. И выводим значения.

* 1. Блок схема алгоритма

Введите элементы массива

i=0; I<MAX\_N; I++

MAX\_N

Начало

Конец

max; min

max=a[i];

a[i]>max

min=i;

a[i]<min

i=1; i<MAX\_N; i++

min = a[0]

max=a[0]

* 1. Листинг программной реализации

#include<stdio.h>

#include<conio.h>

int main()

{

int MAX\_N;

scanf("%d", &MAX\_N);

int a[MAX\_N];// объявление целочисленного массива a из 10 элементов

int i, min, max;

// ввод элементов массива с клавиатуры

for (i=0; i<MAX\_N; i++)

{

printf("Введите элемент mассива %d\n", i);

scanf("%d", &a[i]);

}

min =a[0];// присваиваем минимальному значению значение первого элемента

max=a[0];// присваиваем максимальному значению значение первого элемента

// пробегаем по всем элементам массива

for (i=1; i<MAX\_N; i++)

{

// если текущий элемент массива меньше текущего минимума то присваиваем его значение минимуму

if (a[i]<min)

{

min=i;

}

// если текущий элемент массива больше текущего максимума то присваиваем его значение максимуму

if (a[i]>max)

{

max=a[i];

}

}

// выводим значения на экран

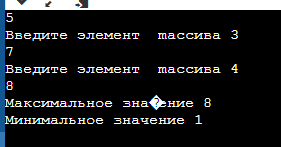
printf("Максимальное значение %d\n",max);

printf("Минимальное значение %d\n",min);

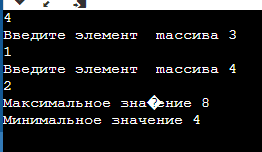
return 0;

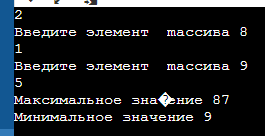
}

* 1. Результат



* 1. Тесты





1. **Постановка задачи №5**

В одномерном массиве, состоящем из п целых элементов, вычислить: сумму модулей элементов массива, расположенных после первого

элемента, равного нулю.

1. **Структурные требования**

Максимальная размерность массива MAX\_N задается глобальной

константой

Текущая размерность массива n вводится в программе через оператор Ввода MAX\_N>0

1. **Описание работы** 
   1. Используемые переменные

a, z, i

* 1. Псевдокод крограммы

Начало

Объявляем массив из 5 int a[5]; элементов. Затем заполняем массив

(i = 0; i<5; i++) через цикл после чего при помощи if ищем ноль и плюсуем следующие

* 1. Блок схема алгоритма

Конец

A, z

i = 0; i<5; i++

z+=fabs(i);

a[i]==0

a

i = 0; i<5; i++

Начало

* 1. Листинг программной реализации

#include <stdio.h>

int main()

{

int a[5]; // объявлен массив a из 5 элементов

int i, z;

// Ввод элементов массива

for (i = 0; i<5; i++)

{

printf("a[%d] = ", i);

scanf("%d", &a[i]); // &a[i] - адрес i-го элемента массива

}

if (a[i]==0)

{

z+=fabs(i);

}

// Вывод элементов массива

for (i = 0; i<5; i++)

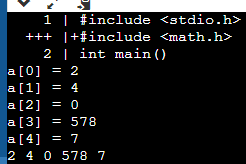
printf("%d ",a[i]); // пробел в формате печати обязателен

//getchar(); getchar();

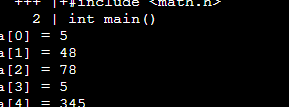
return 0;

}

* 1. Результат



* 1. Тесты



**Список литературы**

* + 1. Конова Е.А., Поллак Г.А. Алгоритмы и программы. Язык С++: Учебное пособие. - 2-е изд., стер. - СПб.: Издательство "Лань", 2017. - 384 с.
    2. Седжвик Роберт. Алгоритмы на С++.: Пер. с англ. - М.: ООО "И.Д. Вильямс", 2011. - 1056 с.
    3. Лафоре Р. Объектно-ориентированное программирование в С++. Классика Computer Science. 4-е изд. - СПб.: Питер, 2015. - 928 с.
    4. Орлов С.А. Теория и практика языков программирования: Учебник для вузов. Стандарт 3-го поколения. - СПб.: Питер, 2014. - 688 с.
    5. Павловская Т.А. С/С++. Процедурное и объектно-ориентированное программирование: Учебник для вузов. Стандарт 3-го поколения. - СПб.: Питер, 2015. - 496 с.
    6. Павловская Т.А. С#. Программирование на языке высокого уровня: Учебник для вузов. - СПб.: Питер, 2014. - 432 с.
    7. Плаксин М.А. Тестирование и отладка программ - для профессионалов будущих и настоящих. - М.: БИНОМ. Лаборатория знаний, 2013. - 167 с.