

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ  
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра прикладної математики

Звіт  
до лабораторної роботи №2  
із дисципліни «Програмування»  
на тему  
ОДНОВИМІРНІ Й ДВОВИМІРНІ МАСИВИ

Виконав:  
студент групи КМ-93  
*Пиндиківський Т. Р.*

Керівник:  
*асистент Дрозденко О. М.*

Київ — 2020

## ЗМІСТ

МЕТА РОБОТИ .....	3
ПОСТАНОВКА ЗАДАЧІ .....	4
ОСНОВНА ЧАСТИНА .....	5
ВИСНОВКИ .....	9
ДОДАТОК А .....	10
ДОДАТОК Б .....	18

## **МЕТА РОБОТИ**

Вивчити базові операції для роботи з одновимірними та двовимірними масивами.

## ПОСТАНОВКА ЗАДАЧІ

### *Завдання 1:*

1. Дано одновимірний масив  $A$ , що складається з  $n$  елементів. Скільки значень елементів зустрічається в масиві по 3 рази?

### *Завдання 2:*

2. Впорядкувати масив методом лінійної вставки.

### *Завдання 3:*

3. Дано матрицю з дійсних чисел. Знайти максимальний і мінімальний елементи й суму елементів, розташованих між ними.

## ОСНОВНА ЧАСТИНА

### Завдання 1 :

Для виконання програми спочатку імпортуються наступні модулі :

- ***stdio.h*** – файл заголовку для стандартних операцій введення/виведення;
- ***time.h*** – бібліотека мови C, що містить функціонал для генерації довільних чисел;

Змінні, що використовуються під час виконання завдання:

- ***length*** (*char*) – рядкова величина, що визначає довжину масиву;
- ***length\_int*** (*int*) – цілочисельна величина, що дорівнює кількості елементів масиву;
- ***number*** (*int*) - число, що прийматиме значення кожного елемента масиву;
- ***counter*** (*int*) - числове значення, що визначає кількість повторень числа у масиві;

Хід виконання завдання:

1. Вводиться значення змінної ***length***. Її введення відбувається у тілі циклу ***do-while***, де умовою виходу з циклу є отримання числового значення довжини масиву, яке не є від'ємним.
2. Відбувається конвертація значення змінної ***length*** у ***length\_int***, яке вже приймає числове значення типу ***int***.
3. Користувачу пропонується вибрати один з способів заповнення масиву числовими значеннями: вручну або довільними числами автоматично.
4. За допомогою двох вкладених циклів ***for*** та змінних ***number***, ***counter*** відбувається підрахування кількості появ одного числа в масиві. Якщо це число – 3, то значення цього елемента масиву виводиться у консоль.

## Завдання 2 :

Для виконання програми спочатку імпортуються наступні модулі :

- ***stdio.h*** – файл заголовку для стандартних операцій введення/виведення;
- ***time.h*** – бібліотека мови C, що містить функціонал для генерації довільних чисел;

Змінні, що використовуються під час виконання завдання:

- ***length (char)*** – рядкова величина, що визначає довжину масиву;
- ***length\_int (int)*** – цілочисельна величина, що дорівнює кількості елементів масиву;
- ***i, j (int)*** – додаткові змінні для ітерування по елементах масиву;

Хід виконання завдання:

1. Вводиться значення змінної ***length***. Її введення відбувається у тілі циклу ***do-while***, де умовою виходу з циклу є отримання числового значення довжини масиву, яке не є від'ємним.
2. Відбувається конвертація значення змінної ***length*** у ***length\_int***, яке вже приймає числове значення типу ***int***.
3. Користувачу пропонується вибрати один з способів заповнення масиву числовими значеннями: вручну або довільними числами автоматично.
4. Потім масив сортується методом вставки, його принцип полягає у наступному :
  - a) Запам'ятати на тимчасову змінну значення поточного елемента масиву;
  - b) Поки елементи зліва від того, що запам'ятав значення більше ніж запам'ятоване – переміщаємо їх на позицію вправо. Виходить, що попередній елемент займе місце запам'ятованого. А той, що стоїть перед попереднім – переміститься в свою чергу на місце попереднього. І так елементи будуть рухатися один за одним.
  - c) Рух елементів закінчується, якщо черговий елемент, який потрібно зрушити, виявляється за значенням менше, ніж той, що запам'ятали в тимчасову змінну на початку циклу.

- d) Цикл бере наступний елемент, і знову зрушує все, які розташовані перед ним і великі за значенням.
4. Відсортований масив виводиться у консоль.
- .

### Завдання 3 :

Для виконання програми спочатку імпортуються наступні модулі :

- ***stdio.h*** – файл заголовку для стандартних операцій введення/виведення;
- ***time.h*** – бібліотека мови C, що містить функціонал для генерації довільних чисел;

Змінні, що використовуються під час виконання завдання:

- ***rows\_str[100]*** (*str*) – рядкова величина, що визначає кількість рядків матриці;
- ***cols\_str[100]*** (*str*) – рядкова величина, що визначає кількість стовпців матриці;
- ***choice\_str[100]*** (*str*) – рядкова величина, що визначає режим заповнення матриці числовими значеннями;
- ***rows\_int*** (*int*) – числова величина, що визначає кількість рядків матриці;
- ***cols\_int*** (*int*) – цілочисельна величина, що визначає кількість стовпців;
- ***choice\_int*** (*int*) – цілочисельне значення, що дорівнює режиму заповнення матриці числами;
- ***max\_value*** (*float*) – число, що визначає значення найбільшого елемента матриці;
- ***min\_value*** (*float*) – число, що визначає значення найменшого елемента матриці;
- ***array*** (*float*) – матриця із чисел.

Хід виконання завдання:

1. Вводиться значення змінної *rows\_str*. Її введення відбувається у тілі циклу *do-while*, де умовою виходу з циклу є отримання числового значення довжини масиву, яке не є від'ємним.
2. Відбувається конвертація значення змінної *rows\_str* у *rows\_int*, яке вже приймає числове значення типу *int*.
3. Вводиться значення змінної *cols\_str*. Її введення відбувається у тілі циклу *do-while*, де умовою виходу з циклу є отримання числового значення довжини масиву, яке не є від'ємним.
4. Відбувається конвертація значення змінної *cols\_str* у *cols\_int*, яке вже приймає числове значення типу *int*.
5. Встановивши мінімальним та максимальним значення першого елемента матриці, відбувається проходження по кожному елементу матриці за допомогою двох вкладених циклів *for*. Якщо на певному етапі значення поточного елемента менше за мінімальне, то мінімальним стає значення поточного елемента. Аналогічно знаходиться максимальне значення елементів матриці. Посилання на адреси мінімального та максимального елементів зберігаються у відповідних вказівниках.
6. Вибравши «меншу» адресу з двох, що посилаються на мінімальний та максимальний елементи, відбувається ітерація по елементах масиву, поки не буде досягнуто значення іншого елемента. Відповідно, всі значення, що лежать між двома вказівниками, сумуються і виводяться в консоль.



## **ВИСНОВКИ**

На цій лабораторній роботі було вивчено базові операції для роботи з одновимірними та двовимірними масивами.

## ДОДАТОК А

### 1. Програмна реалізація задачі №1

```
#include<stdio.h>
#include<time.h>
//#include"isnumber.h"
void program1 ()
{

    srand(time(0));

    char length[100];
    int length_int, i, j;
    do
    {

        printf("\n\nEnter the length of array of numbers : ");
        gets(length);

        } while ((!isintnumber(length)) || (isintnumber(length) &&
!ispositivenumber(length)));

        length_int=atoi(length);
        float array[length_int];

        char choice_str[100];

        do
        {
            printf("\n\nPress 1 - to fill array manually, 2 - to fill with
random numbers : ");
            gets(choice_str);

            } while ((!isintnumber(choice_str)) || (isintnumber(choice_str) &&
!ispositivenumber(choice_str)) || (ispositivenumber(choice_str) && ((atoi(choic
e_str)<1) || (atoi(choice_str)>2)))));

        int choice_int;

        choice_int=atoi(choice_str);
        char number_chr[100];
        switch(choice_int)
        {
            case 2:
            {
                printf("\n\nArray with random numbers :\n\n");
                for (i=0; i<length_int; i++)
                {

                    array[i]= rand () % length_int;
                    printf("%.0f\t", array[i]);

                }
                break;
            }
            case 1:
            {
                for (i=0; i<length_int; i++)
                {
                    do
                    {
```

```

                                printf("\n\nEnter the value of %d element of
array : ", i+1);
                                scanf("%s", &number_chr);
                                } while(!isnumber(number_chr));
                                array[i]=atof(number_chr);
                                }
                                printf("\n\nArray with manually entered numbers :\n\n");
                                for (i=0; i<length_int; i++)
                                    printf("%.0f\t", array[i]);
                                break;
                            }
    }

    int number=0,    counter=0;

    int existence[length_int];
    for(i=0; i<length_int;i++)
        existence[i]=0;

    int k=0, f=0, counter1=0;
    for(i=0; i<length_int;i++)
    {
        counter1=0;
        counter=0;
        number=array[i];

        for(j=i;j<length_int;j++)
        {
            if((existence[j]==0)&&(array[j]==number))
                counter++;
            if(array[j]==number)
                counter1++;
        }

        for(f=0;f<length_int;f++)
        {
            if(array[i]==array[f])
                existence[f]=1;
        }

        if(counter==3)
            printf("\n\nNumber %d meets in array 3 times.", number);
    }
}

```

```

Press the number of task you want to test ( 1 - 3 ) : 1

Task 1 : Print out the elements that occur only 3 times in array .

Enter the length of array of numbers : 30

Press 1 - to fill array manually, 2 - to fill with random numbers : 2

Array with random numbers :
16      27      15      14      26      13      0      19      0      9      14      26      29      9      4
      3      14      10      5      13      18      12      10      19      6      12      3      12      24
      20

Number 14 meets in array 3 times.
Number 12 meets in array 3 times.

```

Рис.1 – Тестування завдання №1

## 2. Програмна реалізація задачі №2

```

#include<stdio.h>
#include<time.h>
//#include"isnumber.h"

void program2 ()
{
    srand(time(0));

    char length[100];
    int length_int, i, j;

    do
    {

        printf("\n\nEnter the length of array of numbers : ");
        gets(length);

        } while ((!isintnumber(length)) || (isintnumber(length) &&
!ispositivenumber(length)));

        length_int=atoi(length);
        float array[length_int];

        char choice_str[100];

        do
        {
            printf("\n\nPress 1 - to fill array manually, 2 - to fill with
random numbers : ");
            gets(choice_str);

            } while ((!isintnumber(choice_str)) || (isintnumber(choice_str) &&
!ispositivenumber(choice_str)) || (ispositivenumber(choice_str) && ((atoi(choic
e_str)<1) || (atoi(choice_str)>2)))));

        int choice_int, c;

```

```

choice_int=atoi(choice_str);
char number_chr[100];
switch(choice_int)
{
    case 2:
    {
        printf("\n\nArray with random numbers :\n\n");
        for (i=0; i<length_int; i++)
        {
            array[i]= rand () % length_int;
            printf("%.0f\t", array[i]);
        }
        break;
    }
    case 1:
    {
        for (i=0; i<length_int; i++)
        {
            do
            {
                printf("\n\nEnter the value of %d element of
array : ", i+1);
                scanf("%s", &number_chr);
            } while(!isnumber(number_chr));
            array[i]=atof(number_chr);
        }
        printf("\n\nArray with manually entered numbers :\n\n");
        for (i=0; i<length_int; i++)
            printf("%.0f\t", array[i]);
        break;
    }
}

for(i=1;i<length_int;i++)
{
    for(j=i;j>0;j--)
    {
        if(array[j]<array[j-1])
        {
            c=array[j-1];
            array[j-1]=array[j];
            array[j]=c;
        }
    }
}

printf("\n\nSorted array: \n\n");
for (i=0; i<length_int; i++)
    printf("%.0f\t", array[i]);
}

```

```

Task 2 : Sort array using selection sort.

Enter the length of array of numbers : 10

Press 1 - to fill array manually, 2 - to fill with random numbers : 2

Array with random numbers :

5      3      0      5      4      1      5      0      7      7

Sorted array:

0      0      1      3      4      5      5      5      7      7

If you want to continue testing program, press c button ... █

```

Рис.2 – Тестування завдання №2

### 3. Програмна реалізація задачі №3

```

#include<stdio.h>
#include<time.h>
//#include"isnumber.h"
void program3 ()
{

    srand(time(0));

    char rows_str[100], cols_str[100];
    int rows_int, cols_int, i, j;
    do
    {

        printf("\n\nEnter the number of rows in a 2d array : ");
        gets(rows_str);

        } while ((!isintnumber(rows_str)) || (isintnumber(rows_str) &&
!ispositivenumber(rows_str)));

    do
    {

        printf("\n\nEnter the number of columns in a 2d array : ");
        gets(cols_str);

        } while ((!isintnumber(cols_str)) || (isintnumber(cols_str) &&
!ispositivenumber(cols_str)));

    rows_int=atoi(rows_str);
    cols_int=atoi(cols_str);

    float array[rows_int][cols_int];

    char choice_str[100];

    do

```

```

{
    printf("\n\nPress 1 - to fill array manually, 2 - to fill with
random numbers : ");
    gets(choice_str);

    } while ((!isintnumber(choice_str)) || (isintnumber(choice_str)    &&
!ispositivenumber(choice_str)) || (ispositivenumber(choice_str) && ((atoi(choic
e_str)<1) || (atoi(choice_str)>2)))));

    int choice_int;
    choice_int=atoi(choice_str);

    char number_chr[100];

    for (i=0; i<rows_int; i++)
        for (j=0; j<cols_int; j++)
            array[i][j]=0;

    int f, k;
    bool result=true;

    switch(choice_int)
    {
        case 1:
        {
            for (i=0; i<rows_int; i++)
            {
                for (j=0; j<cols_int; j++)
                {
                    do
                    {
                        printf("\n\nEnter the value of element of
array, placed in %d row and %d column : ", i, j);
                        gets(number_chr);

                        } while(!isnumber(number_chr));
                        array[i][j]=atof(number_chr);
                    }
                }
            printf("\n\nArray with manually entered numbers :\n");
            for (i=0; i<rows_int; i++)
            {
                printf("\n");
                for (j=0; j<cols_int; j++)
                    printf(" %.2f ", array[i][j]);
                printf("\n");
            }
            break;
        }

        case 2:
        {
            printf("\n\nArray with random numbers :\n");
            for (i=0; i<rows_int; i++)
            {
                printf("\n");
                for (j=0; j<cols_int; j++)
                {
                    do
                    {

```

```

        array[i][j]=          rand          ()          %
(rows_int*cols_int);

        result=true;

        for (k=0; k<i; k++)
        {
            for (f=0; f<cols_int; f++)
            {
                if(array[i][j]==array[k][f])
                {
                    result=false;
                    break;
                }
            }

            if(result==false)
                break;
        }

        for(f=0;f<j;f++)
        {
            if(array[i][j]==array[i][f])
            {
                result&=false;
                break;
            }
        }

        } while(!result);
        printf(" %.2f ", array[i][j]);
    }
    printf("\n");
}
break;
}

float *pointer, *pointer_min, *pointer_max, min_value=array[0][0],
max_value=array[0][0];
pointer=&array[0][0];
float positions[2][2];

for(i=0;i<rows_int*cols_int;pointer++, i++)
{
    if(*pointer<=min_value)
    {
        min_value=*pointer;
        pointer_min=pointer;
    }
    if(*pointer>=max_value)
    {
        max_value=*pointer;
        pointer_max=pointer;
    }
}

printf("\n\nMax value is %.2f .", max_value);
printf("\n\nMin value is %.2f .", min_value);

float sum=0;

```



```

    if(!(pointer_min+1==pointer_max || pointer_max+1==pointer_min))
    {
        if(pointer_min<pointer_max)
        {
            for(pointer=pointer_min, sum=0;
(pointer+1)<pointer_max;sum+=*++pointer /*printf("\nSum   =   %f",   sum),
printf("\nPointer = %p", pointer)*/, pointer++*/);
        }

        if(pointer_min>pointer_max)
        {
            for(pointer=pointer_max, sum=0; (pointer+1)<pointer_min;
sum+=*++pointer/*, printf("\nSum = %f", sum), printf("\nPointer = %p",
pointer)*/, pointer++*/);
        }
    }
    if(sum==0)
        printf("\n\nMin and max values are placed next to each other in
the array.");
    else
        printf("\n\nSum of numbers between min ( %.2f ) and max ( %.2f )
values is : %.2f", min_value, max_value, sum);
}

```

Task 3 : Find min and max values of an array and sum of numbers between min and max values.

Enter the number of rows in a 2d array : 5

Enter the number of columns in a 2d array : 5

Press 1 - to fill array manually, 2 - to fill with random numbers : 2

Array with random numbers :

```

7.00  22.00  20.00  24.00  19.00
23.00  1.00  17.00  4.00  0.00
13.00  8.00  9.00  16.00  2.00
12.00  11.00  3.00  14.00  18.00
21.00  6.00  15.00  10.00  5.00

```

Max value is 24.00 .

Min value is 0.00 .

Sum of numbers between min ( 0.00 ) and max ( 24.00 ) values is : 64.00

If you want to continue testing program, press c button ... ☐

Рис.3 – Тестування завдання №3

## ДОДАТОК Б

### ***Відповіді на запитання:***

*1. Які способи опису масивів застосовують у С?*

Для оголошення масиву використовують такий синтаксис:  $\langle \text{тип даних} \rangle$   
 $\langle \text{ім'я масиву} \rangle [\text{число елементів}];$

*2. Із чого починається індекс?*

Нумерація індексів відбувається з нуля.

*3. Ініціалізація одновимірного масиву.*

```
int powers[4] = { 1, 2, 4, 6 };  
int array_int[100];
```

*4. Звернення до елемента масиву за допомогою посилання.*

Якщо  $\text{pa}$  — посилання, то його можна використовувати з індексом:  $\text{pa}[i]$   
ідентично  $*(\text{pa} + i)$

*5. Отримання адреси й значення елемента багатовимірного масиву.*

$\&\text{array}[i] \sim (\text{pointer} + i)$  — адреса елемента  
 $\text{array}[i] \sim *(\text{pointer} + i)$  — значення елемента

*6. Посилання та масиви.*

Будь-яку дію, яка досягається індексуванням масиву, можна виконати й за допомогою посилань. Варіант із посиланнями буде швидший, але він важчий для розуміння, принаймні, для початківців.

*7. Як отримати доступ до елемента багатовимірного масиву?*

Якщо вміст  $\text{pa}$  вказує на окремий елемент масиву  $\text{a}$ , то за визначенням  $\text{pa} + 1$  вказує на наступний елемент,  $\text{pa} - i$  — на  $i$ -й елемент перед  $\text{pa}$ ,  $\text{pa} + i$  — на  $i$ -й елемент після  $\text{pa}$ . Таким чином, якщо  $\text{pa}$  вказує на  $\text{a}[0]$ , то  $*(\text{pa} + 1)$  відповідає вмісту  $\text{a}[1]$ ,  $\text{pa} + i$  є адресою  $\text{a}[i]$ , а  $*(\text{pa} + i)$  — вмістом  $\text{a}[i]$ .