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: | Керівник: |
| студент групи КМ-93 | *асистент Дрозденко О. М.* |
| *Пиндиківський Т. Р.* |  |

Київ — 2020

ЗМІСТ

[МЕТА РОБОТИ 3](#_Toc34081176)

[ПОСТАНОВКА ЗАДАЧІ 4](#_Toc34081177)

[ОСНОВНА ЧАСТИНА 5](#_Toc34081178)

[ВИСНОВКИ 12](#_Toc34081179)

[ДОДАТОК А 13](#_Toc34081180)

[ДОДАТОК Б 21](#_Toc34081181)

# 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 інформацію про довжину масиву;
* ***length\_int*** (*int*) –цілочисельна величина, що дорівнює кількості елементів масиву;
* ***number*** (*int*) - число, що приймає значення кожного елемента масиву;
* ***counter*** (*int*) - числове значення, що визначає кількість повторень числа у масиві;
* ***array[]*** (*int*) – масив чисел типу int.

Хід виконання завдання:

1. Вводиться значення змінної ***length*** з використанням функції ***gets()***, що повертає введену користувачем рядкову величину*.* Її введення відбувається у тілі циклу***do-while***, де умовою виходу з циклу є отримання рядкової величини, що по суті є числом - довжини масиву, яке не є від’ємним***((!isintnumber(length))||(isintnumber(length)&&!ispositivenumber(length)))***.
2. Відбувається конвертація значення змінної ***length*** у***length\_int*** з використанням стандартної функції ***atoi()***, що конвертує рядкову величину у цілочисельну, тому змінна ***length\_int*** вже прийматиме значення типу ***int***.
3. Користувачу виводиться повідомлення на екран про можливість вибору заповнення масиву числовими значеннями: вручну або довільними числами автоматично. Потім користувачем вводиться значення «1» або «2» з використанням функції ***scanf()*** із специфікаторм типу ***“%s”*** та подальшою конвертацією значення змінної ***choice*** у цілочисельний тип із застосуванням функції ***atoi()***.
4. При виборі опції заповнення масиву довільними значеннями «запускається» цикл ***for()*** із подальшою ітерацією по значенню змінної ***i*** ( від 0 до («довжини масиву»-1)) та присвоєнню кожному елементу масиву ***array[i]*** довільного значення, що отримується при виклику функції ***rand().***
5. При виборі опції заповнення масиву вручну «запускається» цикл ***for()*** із подальшою ітерацією по значенню змінної ***i*** ( від 0 до («довжини масиву»-1)) та присвоєнню кожному елементу масиву ***array[i]*** значення, отримується при виклику функції ***scanf(“%s”, &array[i])*** та подальшою конвертацією у числове значення (із застосуванням функції ***atoi()***).
6. За допомогою циклу ***for()***, що ітерує по елементах масиву ***array[i]***,де ***i*** змінюється від 0 до («довжини масиву»-1). Із використаннням змінної цілого типу counter відбувається підрахування кількості появ одного числа в масиві – додається 1, якщо значення виразу (***array[i]==number***) є істинним твердженням. Якщо після виконання команд циклу значення змінної ***counter*** - число 3, то елемент масиву виводиться у консоль з використанням функції ***printf(“%d”, number[i])***.

***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;
* ***choice*** (*int*) – змінна, у якій зберігається значення «вибору» користувача типу/методу заповнення масиву числовими значеннями.

Хід виконання завдання:

1. Вводиться значення змінної ***length*** з використанням функції ***gets()***, що повертає введену користувачем рядкову величину. Її введення відбувається у тілі циклу ***do-while()***, де умовою виходу з циклу є отримання рядкової величини, що по суті є числом - довжини масиву, яке не є від’ємним ***((!isintnumber(length))||(isintnumber(length)&&!ispositivenumber(length))).***

2. Відбувається конвертація значення змінної ***length*** у ***length\_int*** з використанням стандартної функції ***atoi(),*** що конвертує рядкову величину у цілочисельну, тому змінна ***length\_int*** вже прийматиме значення типу ***int***.

3. Користувачу виводиться повідомлення на екран про можливість вибору заповнення масиву числовими значеннями: вручну або довільними числами автоматично. Потім користувачем вводиться значення «1» або «2» з використанням функції ***scanf()*** із специфікаторм типу ***“%s”*** та подальшою конвертацією значення змінної ***choice*** у цілочисельний тип із застосуванням функції ***atoi().***

4. При виборі опції заповнення масиву довільними значеннями «запускається» цикл ***for()*** із подальшою ітерацією по значенню змінної ***i*** ( від 0 до («довжини масиву»-1)) та присвоєнню кожному елементу масиву ***array[i]*** довільного значення, що отримується при виклику функції ***rand().***

5. При виборі опції заповнення масиву вручну «запускається» цикл ***for()*** із подальшою ітерацією по значенню змінної ***i*** ( від 0 до («довжини масиву»-1)) та присвоєнню кожному елементу масиву ***array[i]*** значення, отримується при виклику функції ***scanf(“%s”, &array[i])*** та подальшою конвертацією у числове значення (із застосуванням функції ***atoi()***).

6. Потім масив сортується методом вставки, його принцип полягає у наступному :

1. Запам'ятати на тимчасову змінну значення поточного елемента масиву;
2. 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.
3. Рух елементів закінчується, якщо черговий елемент, який потрібно зрушити, виявляється за значенням менше, ніж той, що запам'ятали в тимчасову змінну на початку циклу.
4. Цикл бере наступний елемент, і знову зрушує все, які розташовані перед ним і великі за значенням.
5. Відсортований масив виводиться у консоль: для цього «запускається» цикл ***for()*** із подальшою ітерацією по значенню змінної ***i*** ( від 0 до («довжини масиву»-1)) та виведення кожного елементу масиву ***array[i]*** із використанням функції ***printf(“%d”, array[i]).***

.

***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 мариці;
* ***min\_value*** (*float*) - число, що визначає значення найменшого елемента матриці;
* ***array[][]*** (*float*) – матриця із чисел.

Хід виконання завдання:

1. Вводиться значення змінної ***rows\_str*** з використанням функції ***gets(),*** що повертає введену користувачем рядкову величину. Її введення відбувається у тілі циклу ***do-while(),*** де умовою виходу з циклу є отримання рядкової величини, що по суті є числом – кількості рядків масиву, яке не є від’ємним ***((!isintnumber(rows\_str))||(isintnumber(rows\_str)&&!ispositivenumber(rows\_str))).***
2. Відбувається конвертація значення змінної ***rows\_str*** у ***rows\_int***, яке вже приймає числове значення типу ***int*** за допомогою функції ***atoi(),*** що конвертує дані типу ***char*** у тип ***int*** .
3. Вводиться значення змінної ***cols\_str*** з використанням функції ***gets()***, що повертає введену користувачем рядкову величину. Її введення відбувається у тілі циклу ***do-while(),*** де умовою виходу з циклу є отримання рядкової величини, що є числом – кількості стовпців масиву, яке є цілим та додатнім ***((!isintnumber(cols\_str))||(isintnumber(cols\_str)&&!ispositivenumber(cols\_str))).***
4. Відбувається конвертація значення змінної ***cols\_str*** у ***cols\_int***, яке вже приймає числове значення типу ***int*** за допомогою функції ***atoi()***, що конвертує дані типу ***char*** у тип ***int*** .
5. Встановивши мінімальним та максимальним значення першого елемента матриці, відбувається проходження по кожному елементу матриці за допомогою двох вкладених циклів ***for*** (відбувається ітерація по змінній ***i,*** що приймає значення від 0 до («кількості рядків»-1) та по змінній ***j***, що приймає значення від 0 до («кількості стовпців»-1). Якщо на певній ітерації значення поточного елемента ***array[i][j]*** менше за мінімальне ***(array[i][j]<min\_value),*** то мінімальним стає значення поточного елемента. Аналогічно знаходиться максимальне значення елементів матриці. Після цього запам’ятовуються адреси мінімального та максимального елементів двовимірного масиву з використанням вказівників ***\*pointer\_min*** та ***\*pointer\_max.*** Посилання на адреси мінімального та максимального елментів зберігаються у відповідних вказівниках. Після цього відбувається ітерація циклу ***for()*** по значеннях адрес вказівників від мінімального до максимального елементів ***for(pointer=pointer\_min;pointer+1<pointer\_max;pointer++)*** та послідовним сумуванням у тілі циклу кожного значення елемента матриці, що відповідає значенню вказівника ***\*pointer – sum+=\*pointer;***
6. В консоль виводиться значення змінної ***sum*** з вкористанням функції ***printf(“%s”, sum).***

# 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(choice\_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);

}

}

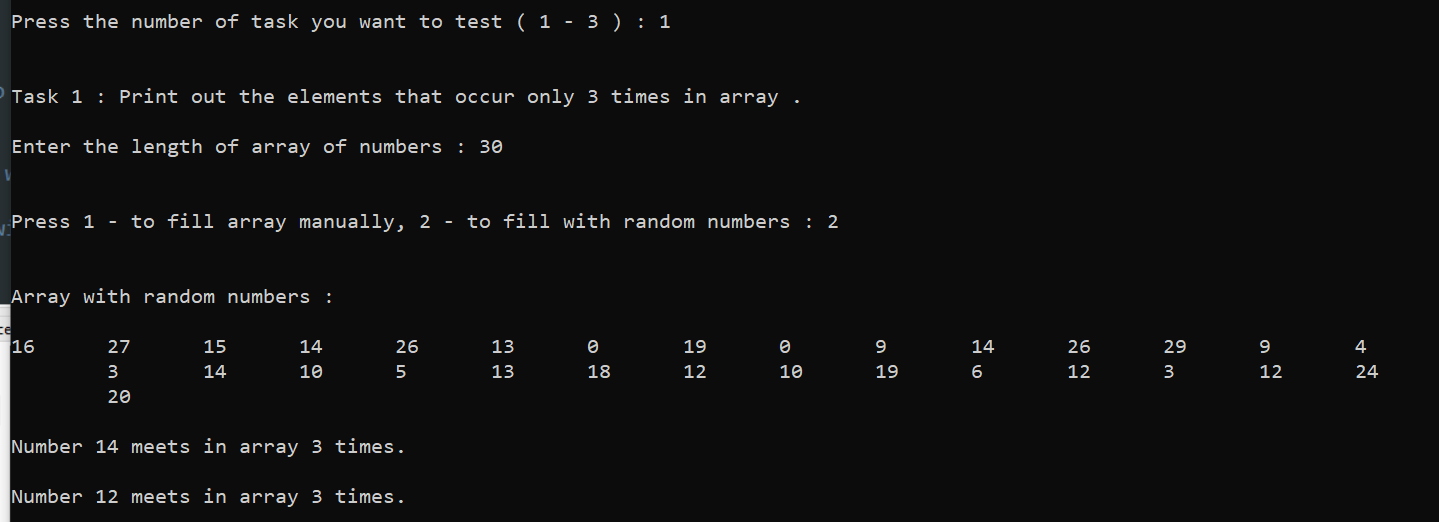


Рис.1 – Тестування завдання №1

1. ***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(choice\_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]);

}

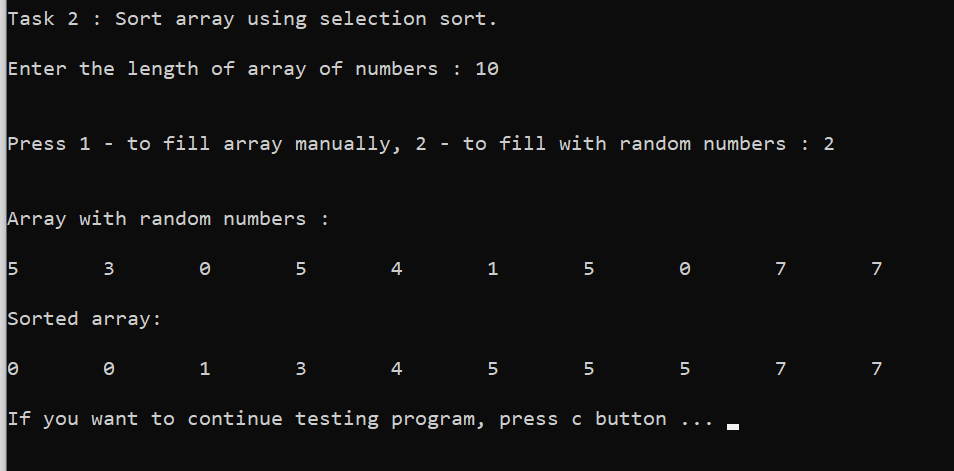
******

Рис.2 – Тестування завдання №2

1. ***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(choice\_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);

}

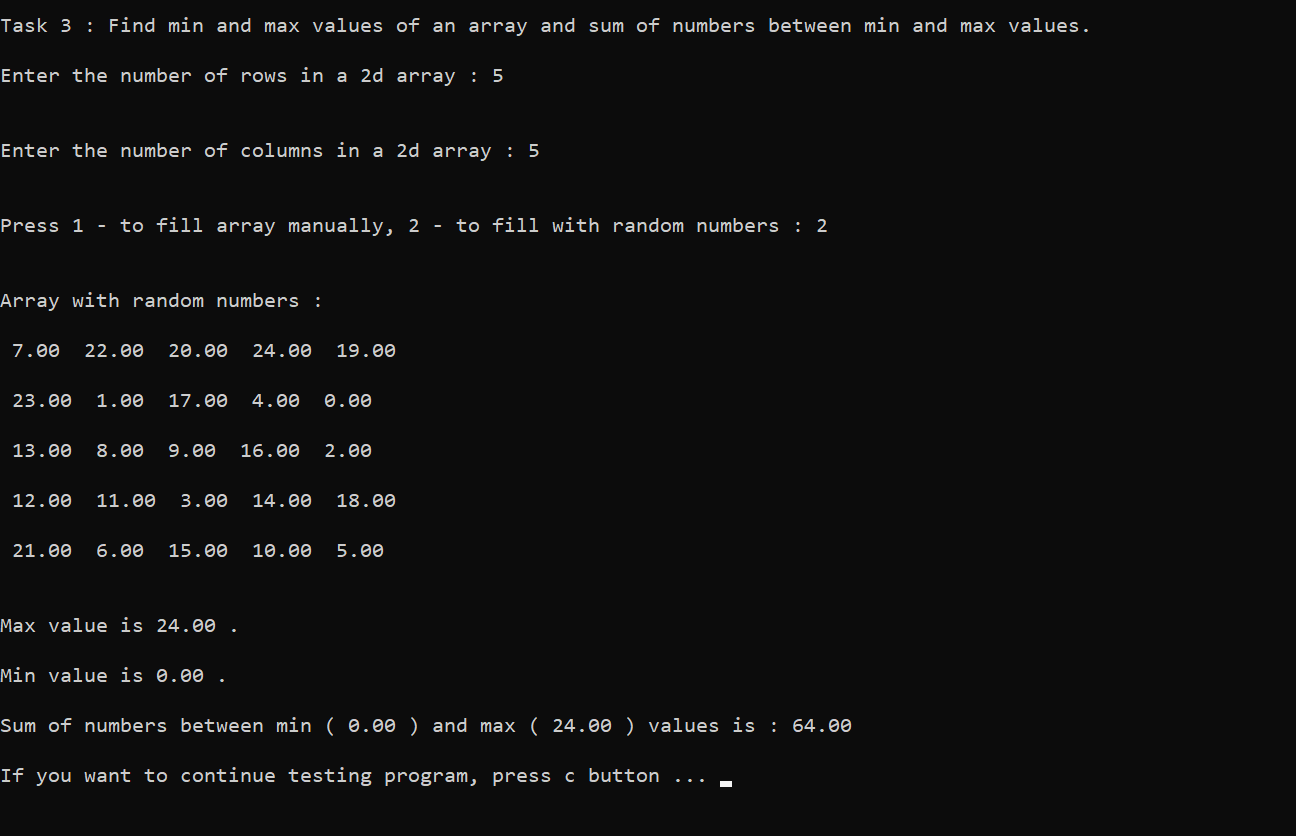
******

Рис.3 – Тестування завдання №3

# ДОДАТОК Б

***Відповіді на запитання:***

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

Для оголошення масиву використовують такий синтаксис: <тип даних> <ім’я масиву> [число елементів];

1. *Із чого починається індекс?*

Нумерація індексів відбувається з нуля.

1. *Ініціалізація одновимірного масиву.*

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

int array\_int[100];

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

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

1. *Отримання адреси й значення елемента багатовимірного массиву.*

&array[i] ~ (pointer + i) – адреса елемента

array[i] ~ \*(pointer + i) – значення елемента

1. *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.

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

Якщо вміст pa вказує на окремий елемент масиву a, то за визначен-ням pa + 1 указує на наступний елемент, pa - i — на i-й елемент пе-ред pa, pa + i — на i-й елемент після pa. Таким чином, якщо pa вказує на a[0], то \*(pa + 1) відповідає вмісту a[1], pa + i є адресою a[i], a \*(pa + i) — вмістом a[i].