rr

**МIНIСТЕРСТВО ОСВIТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**

**ІМЕНІ ІГОРЯ СІКОРСЬКОГО”**

**Факультет прикладної математики**

**Кафедра програмного забезпечення комп’ютерних систем**

**Лабораторна робота № 4**

**з дисципліни “ Основи програмування ”**

**тема “ОДНОВИМІРНІ ТА ДВОВИМІРНІ МАСИВИ ДАНИХ”**

|  |  |  |
| --- | --- | --- |
| **Виконав**  **студент I курсу**  **групи КП-83**  **Мортіков Владислав Євгенович**  **варіант №16** |  | **Перевірив**  **“\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.**  **викладач**  **Гадиняк Руслан Анатолійович**  **(прізвище, ім’я, по батькові)** |

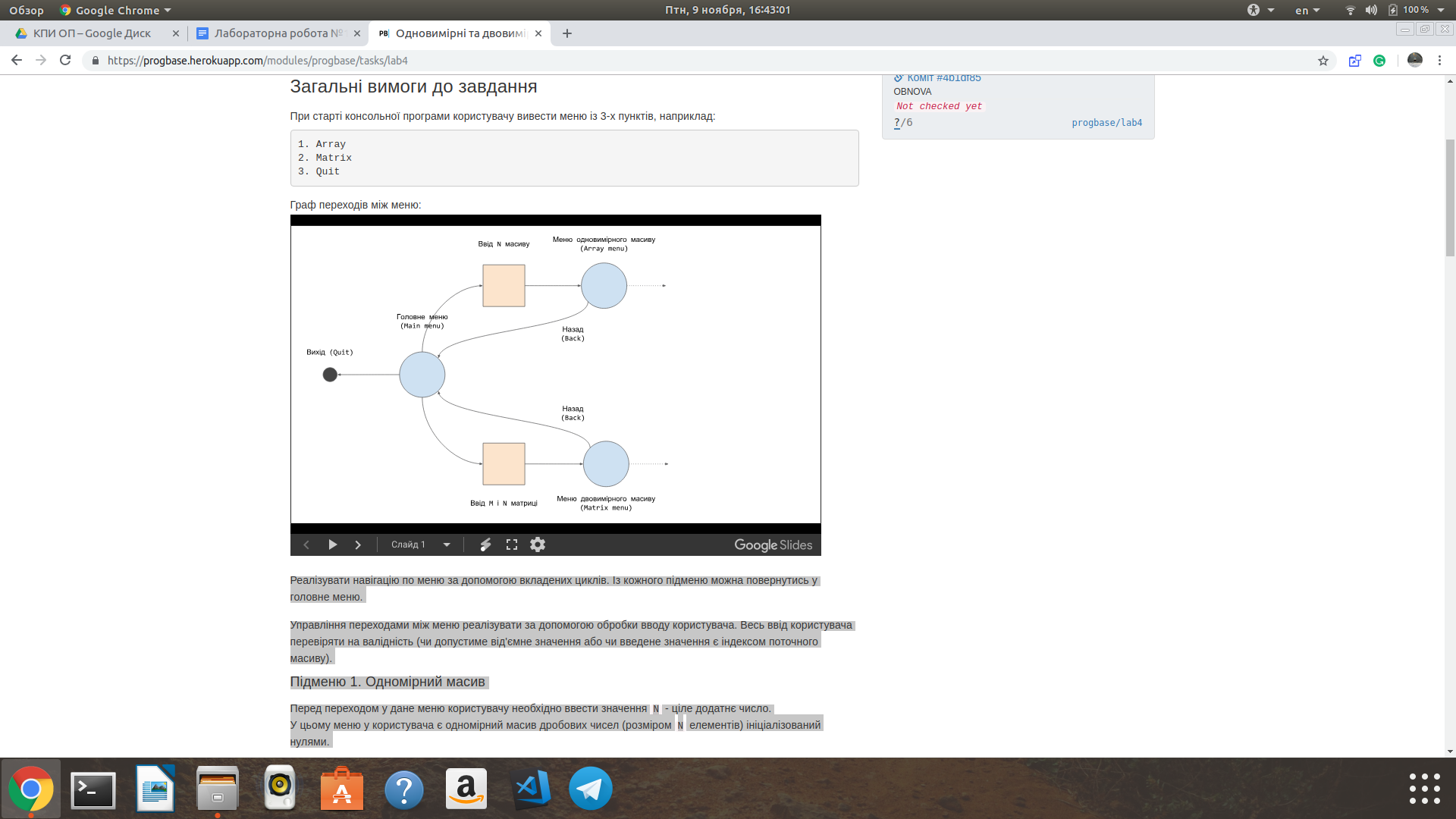
**Київ 2018**

**Мета роботи**

Навчитися працювати із масивами різних типів даних та розмірностей. Застосувати на практиці різні види циклічних конструкцій при роботі з одновимірними та багатовимірними масивами даних. Навчитися оформлювати консольну програму для зручності роботи користувача.

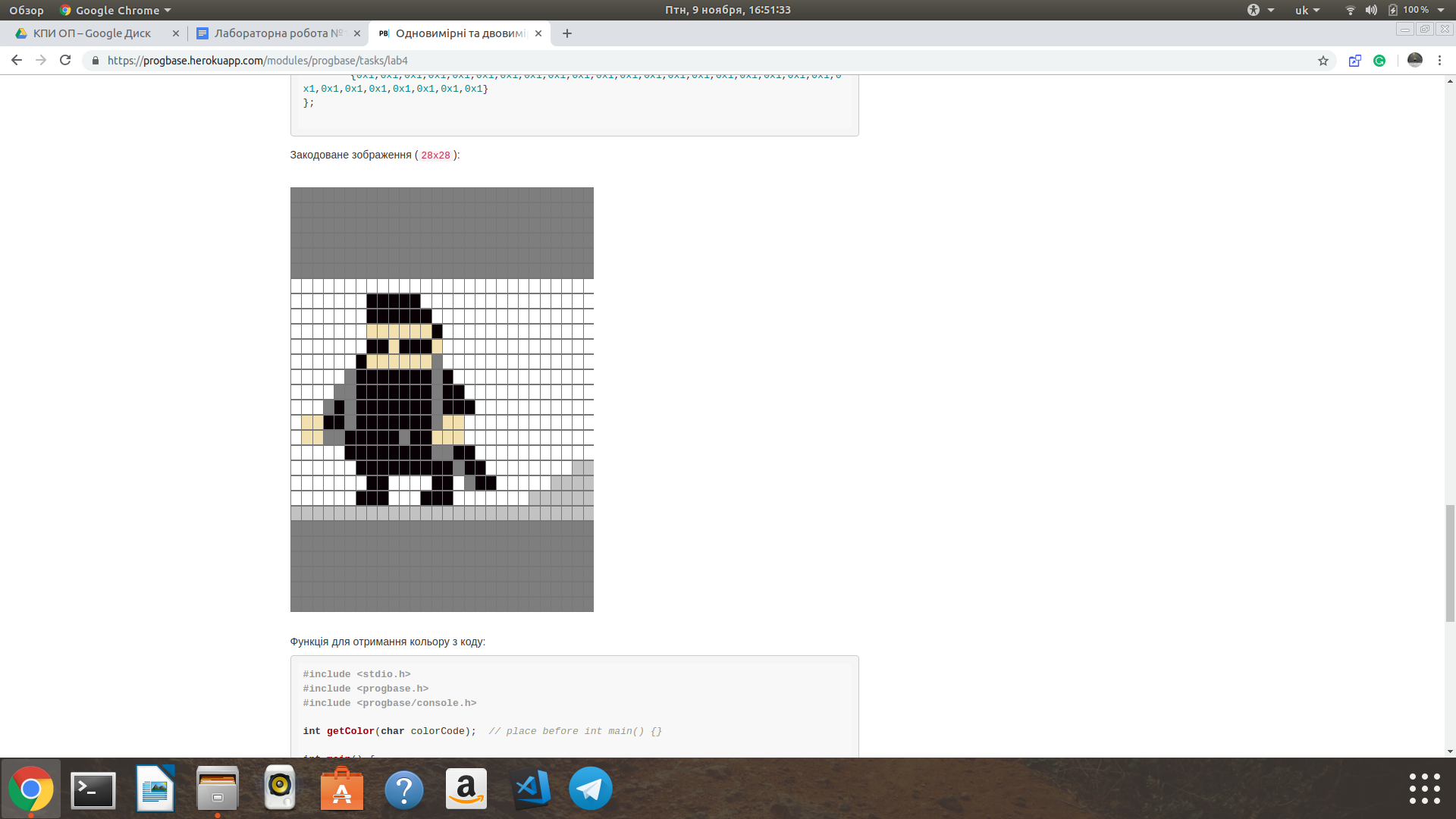
**Постановка завдання**

При старті консольної програми користувачу вивести меню із 3-х пунктів, наприклад:  
 ***1 >> Array  
 2 >> Matrix  
 0 >> Quit***

Реалізувати навігацію по меню за допомогою вкладених циклів. Із кожного підменю можна повернутись у головне меню.  
 Управління переходами між меню реалізувати за допомогою обробки вводу користувача. Весь ввід користувача перевіряти на валідність (чи допустиме від'ємне значення або чи введене значення є індексом поточного масиву).

***Підменю 1. Одномірний масив***  
 Перед переходом у дане меню користувачу необхідно ввести значення N - ціле додатнє число. У цьому меню у користувача є одномірний масив дробових чисел *(розміром N елементів)* ініціалізований нулями.  
 Вивести масив у консоль та запропонувати користувачу такі операції над ним:  
 ***1 >> Заповнити масив випадковими числами від L до H.  
 2 >> Обнулити всі елементи масиву.  
 3 >> Знайти максимальний елемент масиву та його індекс.  
 4 >> Вивести суму додатніх елементів масиву.  
 5 >> Поміняти місцями значення макс і мін елементів масиву.  
 6 >> Збільшити всі елементи масиву на введене число.***

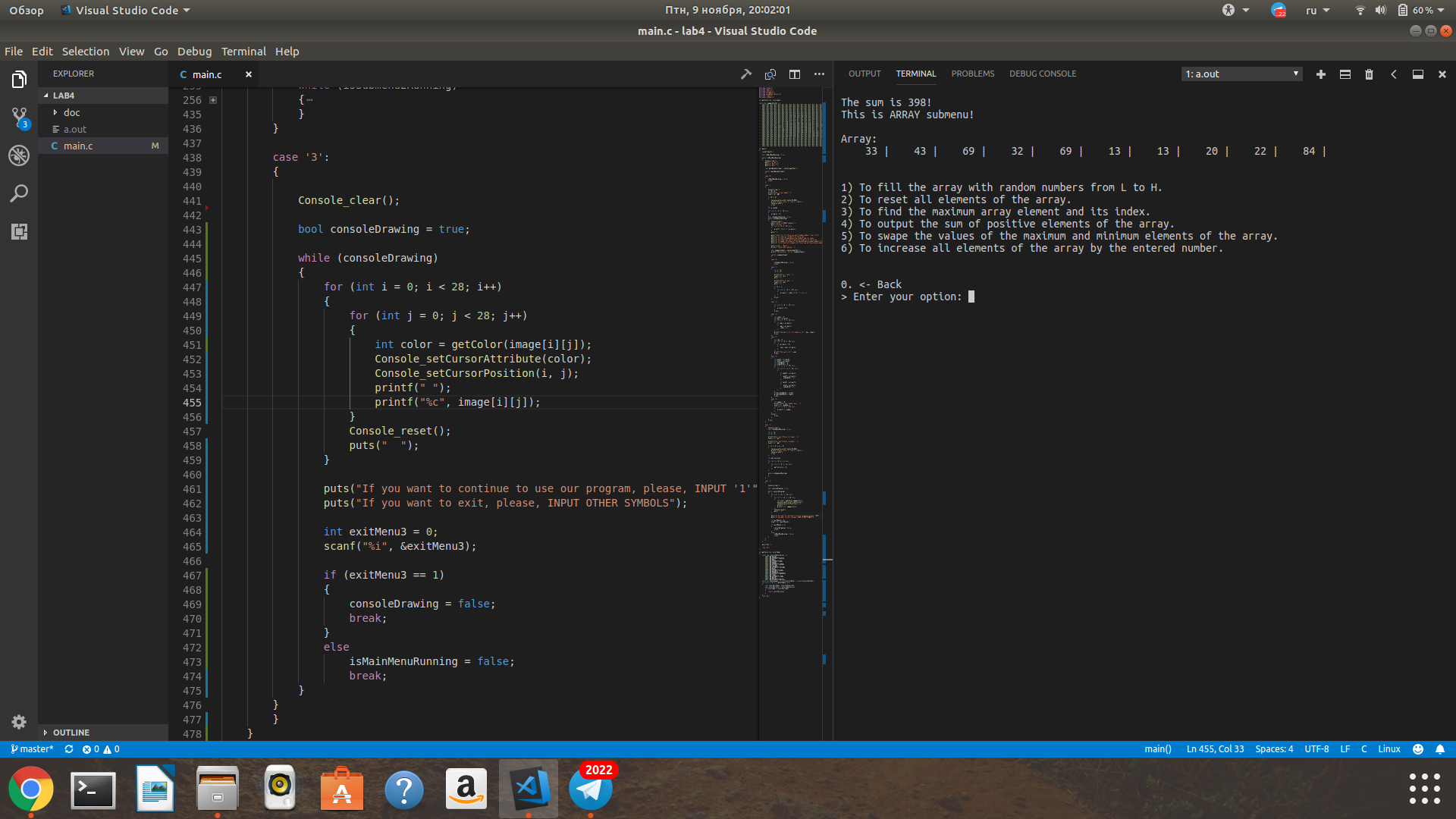
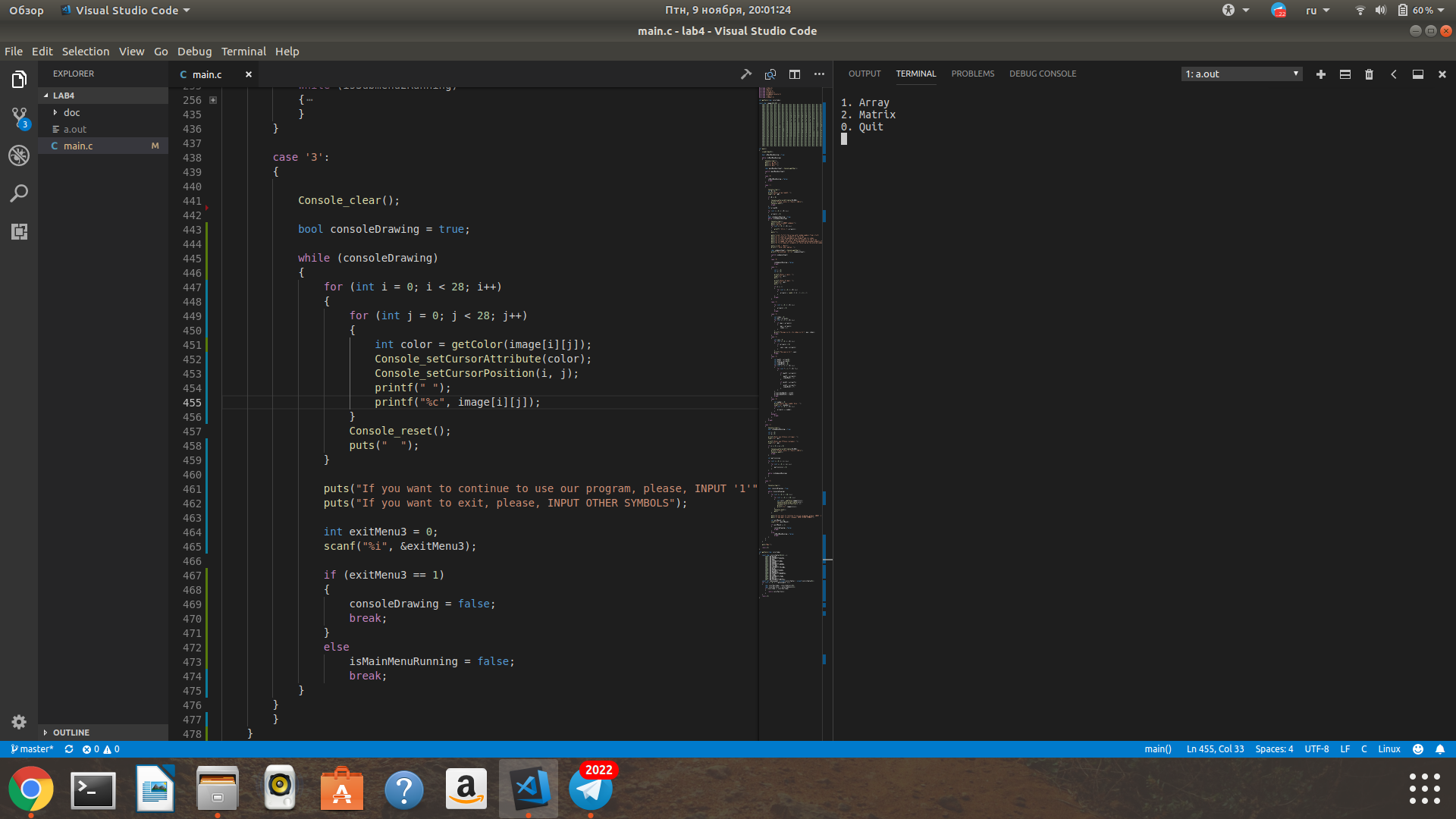
***Підменю 2. Двовимірний масив*** Перед переходом у дане меню користувачу необхідно ввести значення N та M - цілі додатні числа. У цьому меню у користувача є двовимірний масив цілих чисел *(матриця NхM)* ініціалізований нулями.  
 Вивести матрицю у консоль та запропонувати користувачу такі операції над нею:  
 ***1 >> Заповнити масив випадковими числами від L до H.  
 2 >> Обнулити всі елементи масиву.  
 3 >> Знайти мінімальний елемент та його індекси (i та j).  
 4 >> Знайти суму елементів рядка за заданим індексом.  
 5 >> Поміняти місцями макс і мін елементи масиву.  
 6 >> Змінити значення ел. за вказаними індексами на задане.***  
 ***Підменю 2+3/4 (Бонус)***. *Цей пункт не є обов'язковим.*  
 У додатках завдання задано матрицю кодів кольорів (можна розмістити її як глобальну змінну) та функцію для отримання кольору на основі коду цього кольору. У даному підменю вивести зображення, що закодоване, у консоль.

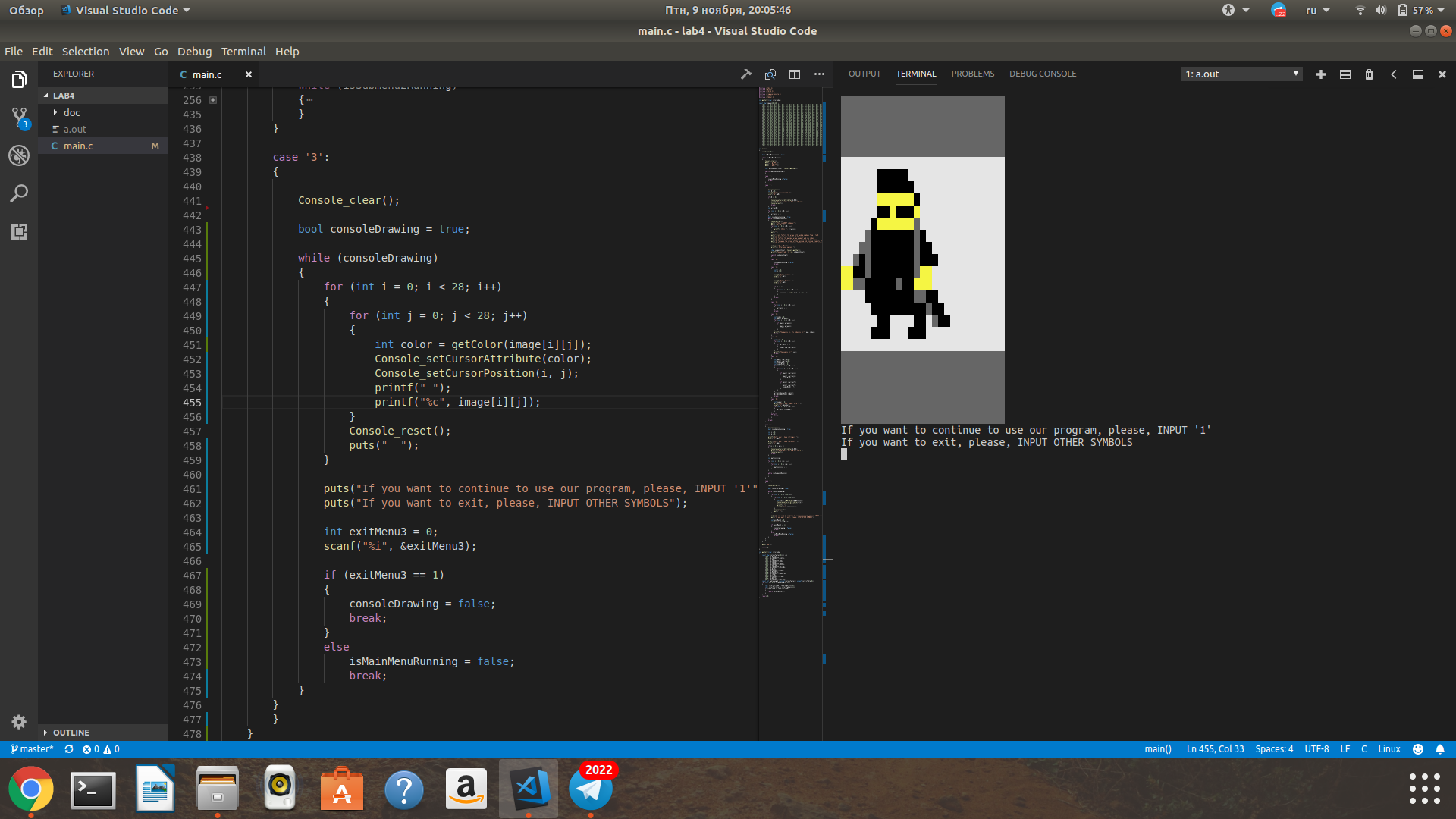


**Тексти коду програм**

|  |
| --- |
| **main.c** |
| #include <stdio.h>  #include <math.h>  #include <stdlib.h>  #include <progbase.h>  #include <progbase/console.h>  #include <time.h>  #include <stdbool.h>  int getColor(char colorCode);  const char image[28][28] = {  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0x0, 0x0, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x7, 0x7, 0x7, 0x7, 0x7, 0x7, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0x7, 0x0, 0x0, 0x0, 0x7, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x7, 0x7, 0x7, 0x7, 0x7, 0x7, 0x1, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0x1, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x1, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0x1, 0x1, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x1, 0x0, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0x1, 0x0, 0x1, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x1, 0x0, 0x0, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0x7, 0x7, 0x0, 0x0, 0x1, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x1, 0x7, 0x7, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0x7, 0x7, 0x1, 0x1, 0x0, 0x0, 0x0, 0x0, 0x0, 0x1, 0x0, 0x0, 0x7, 0x7, 0x7, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x1, 0x1, 0x0, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x1, 0x0, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xE, 0xE},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0xF, 0x1, 0x0, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xE, 0xE, 0xE, 0xE},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0x0, 0xF, 0xF, 0xF, 0x0, 0x0, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE},  {0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE, 0xE},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1}};  int main()  {  srand(time(0));  bool isMainMenuRunning = true;  while (isMainMenuRunning)  {  Console\_clear();  puts("1. Array ");  puts("2. Matrix");  puts("0. Quit ");  char mainMenuUserInput = Console\_getChar();  switch (mainMenuUserInput)  {  case '0':  {  isMainMenuRunning = false;  break;  }  case '1':  {  Console\_clear();  int N = 0;  printf("Enter array length: ");  scanf("%d", &N);  if (N <= 0)  {  Console\_setCursorAttribute(FG\_RED);  printf("Please, enter n > 0 & n < 100\n");  Console\_reset();  break;  }  int array[N];  for (int i = 0; i < N; i++)  {  array[i] = 0;  }  bool isSubmenu1Running = true;  while (isSubmenu1Running)  {  Console\_clear();  puts("\nThis is ARRAY submenu!");  puts("\nArray: ");  for (int i = 0; i < N; i++)  {  printf(" %5.1i |", array[i]);  }  puts("");  puts("\n\n1) To fill the array with random numbers from L to H. ");  puts("2) To reset all elements of the array. ");  puts("3) To find the maximum array element and its index. ");  puts("4) To output the sum of positive elements of the array. ");  puts("5) To swape the values of the maximum and minimum elements of the array.");  puts("6) To increase all elements of the array by the entered number. ");  puts("\n\n0. <- Back");  printf("> Enter your option: ");  char submenu1Input = Console\_getChar();  printf("You entered: '%c'\n", submenu1Input);  switch (submenu1Input)  {  case '0':  isSubmenu1Running = false;  break;  case '1':  {  int l = 0;  int h = 0;  printf("Enter L (min): ");  scanf("%i", &l);  puts(" ");  printf("Enter H (max): ");  scanf("%i", &h);  puts(" ");  if (h >= l)  {  for (int i = 0; i < N; i++)  {  array[i] = rand() % (h - l + 1) + l;  }  }  break;  }  case '2':  for (int i = 0; i < N; i++)  {  array[i] = 0;  }  break;  case '3':  {  int index = 0;  int max = array[0];  for (int i = 1; i < N; i++)  {  if (max < array[i])  {  max = array[i];  index = i;  }  }  printf("The max is %i, its index is %i!", max, index);  break;  }  case '4':  {  int sum = 0;  for (int i = 0; i < N; i++)  {  if (array[i] > 0)  {  sum = sum + array[i];  }  }  printf("The sum is %i!", sum);  break;  }  case '5':  {  int maxEl = array[0];  int minEl = array[0];  int indexMaxEl = 0;  int indexMinEl = 0;  for (int i = 1; i < N; i++)  {  for (int f = 1; f < N; f++)  {  if (maxEl < array[i])  {  maxEl = array[i];  indexMaxEl = i;  }  if (minEl > array[f])  {  minEl = array[f];  indexMinEl = f;  }  }  }  array[indexMaxEl] = minEl;  array[indexMinEl] = maxEl;  break;  }  case '6':  {  int number = 0;  printf("Enter your number here - ");  scanf("%i", &number);  for (int i = 0; i < N; i++)  {  array[i] += number;  }  }  default:  break;  }  }  break;  }  case '2':  {  Console\_clear();  bool isSubmenu2Running = true;  int n = 0;  int m = 0;  printf("Enter your N here (strings): ");  scanf("%i", &n);  printf("Enter your M here (columns): ");  scanf("%i", &m);  if (n <= 0 || m <= 0)  {  Console\_setCursorAttribute(FG\_RED);  printf("Please, enter n > 0 & n < 100\n");  Console\_reset();  break;  }  int matrix[n][m];  for (int i = 0; i < n; i++)  {  for (int j = 0; j < m; j++)  {  matrix[i][j] = 0;  }  }  while (isSubmenu2Running)  {  Console\_clear();  puts("\nThis is MATRIX submenu!");  puts("\nMatrix: ");  for (int i = 0; i < n; i++)  {  for (int j = 0; j < m; j++)  {  printf(" %5.1i |", matrix[i][j]);  }  puts(" ");  }  puts("\n\n1) To fill the matrix with random numbers from L to H.");  puts("2) To reset all elements of the array.");  puts("3) To find the minimal element and its indexes (i and j).");  puts("4) To find the sum of line items for a given index.");  puts("5) To swape places of the maximum and minimum elements of the array.");  puts("6) To change the value of an item by specified indexes to a specified one.");  puts("\n\n0. <- Back");  printf("> Enter your option: ");  char submenu2Input = Console\_getChar();  printf("You entered: '%c'\n", submenu2Input);  switch (submenu2Input)  {  case '0':  {  isSubmenu2Running = false;  break;  }  case '1':  {  int a = 0;  int b = 0;  printf("Enter L (min): ");  scanf("%i", &a);  puts(" ");  printf("Enter H (max): ");  scanf("%i", &b);  puts(" ");  for (int i = 0; i < n; i++)  {  for (int j = 0; j < m; j++)  {  matrix[i][j] = rand() % (b - a + 1) + a;  }  }  break;  }  case '2':  {  for (int i = 0; i < n; i++)  {  for (int j = 0; j < m; j++)  {  matrix[i][j] = 0;  }  }  }  case '3':  {  int indexI = 0;  int indexJ = 0;  int min = matrix[0][0];  for (int i = 1; i < n; i++)  {  for (int j = 1; j < m; j++)  {  if (min > matrix[i][j])  {  min = matrix[i][j];  indexI = i;  indexJ = j;  }  }  }  printf("The min is %i, its index is [%2.1i][%2.1i]!", min, indexI, indexJ);  break;  }  case '4':  {  int sum = 0;  int i = 0;  printf("> I want to use this line - ");  scanf("%i", &i);  if (i >= 0 && i < n)  {  for (int j = 0; j < m; j++)  {  sum = sum + matrix[i][j];  }  printf("The sum of elements when i = %i is %i", i, sum);  }  else  printf("Please, print another i here");  break;  }  case '5':  {  int maxEl = matrix[0][0];  int minEl = matrix[0][0];  int indexIMaxEl = 0;  int indexJMaxEl = 0;  int indexIMinEl = 0;  int indexJMinEl = 0;  for (int i = 1; i < n; i++)  {  for (int j = 1; j < m; j++)  {  for (int g = 1; g < n; g++)  {  for (int f = 1; f < m; f++)  {  if (maxEl < matrix[i][j])  {  maxEl = matrix[i][j];  indexIMaxEl = i;  indexJMaxEl = j;  }  if (minEl > matrix[g][f])  {  minEl = matrix[g][f];  indexIMinEl = g;  indexJMinEl = f;  }  }  }  }  }  matrix[indexIMaxEl][indexJMaxEl] = minEl;  matrix[indexIMinEl][indexJMinEl] = maxEl;  break;  }  case '6':  {  int i = 0;  int j = 0;  float indexValue = 0;  printf("Enter your i here - ");  scanf("%i", &i);  printf("Enter your j here - ");  scanf("%i", &j);  if (i >= 0 && i < n && j >= 0 && j < m)  {  printf("\n\nelement[%i][%i] = %i", i, j, matrix[i][j]);  puts("\nWhat value do you want?");  printf("\n\n> I want its ");  scanf("%f", &indexValue);  matrix[i][j] = indexValue;  }  else  printf("Please, input valid dates!");  break;  }  default:  {  }  }  }  }  case '3':  {  Console\_clear();  bool consoleDrawing = true;  while (consoleDrawing)  {  for (int i = 0; i < 28; i++)  {  for (int j = 0; j < 28; j++)  {  int color = getColor(image[i][j]);  Console\_setCursorAttribute(color);  Console\_setCursorPosition(i, j);  printf(" ");  printf(" %c ", image[i][j]);  }  Console\_reset();  puts(" ");  }  puts("If you want to continue to use our program, please, INPUT '1'");  puts("If you want to exit, please, INPUT OTHER SYMBOLS");  int exitMenu3 = 0;  scanf("%i", &exitMenu3);  if (exitMenu3 == 1)  {  consoleDrawing = false;  break;  }  else  isMainMenuRunning = false;  break;  }  }  }  }  puts("Bye!");  return 0;  }  int getColor(char colorCode)  {  const char colorsTable[16][2] = {  {0x0, BG\_BLACK},  {0x1, BG\_INTENSITY\_BLACK},  {0x2, BG\_RED},  {0x3, BG\_INTENSITY\_RED},  {0x4, BG\_GREEN},  {0x5, BG\_INTENSITY\_GREEN},  {0x6, BG\_YELLOW},  {0x7, BG\_INTENSITY\_YELLOW},  {0x8, BG\_BLUE},  {0x9, BG\_INTENSITY\_BLUE},  {0xa, BG\_MAGENTA},  {0xb, BG\_INTENSITY\_MAGENTA},  {0xc, BG\_CYAN},  {0xd, BG\_INTENSITY\_CYAN},  {0xe, BG\_WHITE},  {0xf, BG\_INTENSITY\_WHITE}};  const int tableLength = sizeof(colorsTable) / sizeof(colorsTable[0]);  for (int i = 0; i < tableLength; i++)  {  char colorPairCode = colorsTable[i][0];  char colorPairColor = colorsTable[i][1];  if (colorCode == colorPairCode)  {  return colorPairColor;  }  }  return 0;  } |

**￼Приклади результатів**





**Висновки**

Виконавши дану лабораторну роботу ми навчилися працювати із масивами різних типів даних та розмірностей. Застосували на практиці різні види циклічних конструкцій (з передумовою, із лічильником) при роботі з одновимірними та багатовимірними масивами даних.

Консольна програма оформлена для зручності роботи користувача.

Компіляція всього коду відбувалася за допомогою компілятора gcc.