

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(ла)  студент(ка) I курсу  групи КП-83  Клименко Ярослав Олександрович  (*прізвище, ім’я, по батькові*)    варіант № 8 |  | Перевірив  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладач  Гадиняк Руслан Анатолійович  (*прізвище, ім’я, по батькові*) |

Київ 2018

**Мета роботи**

Навчитися працювати із масивами різних типів даних та розмірності.

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.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 максимального і мінімального елементів масиву.
6. Збільшити всі елементи масиву на введене число.

#### Підменю 2. Двомірний масив

Перед переходом у дане меню користувачу необхідно ввести значення N та M - цілі додатні числа.

У цьому меню у користувача є двомірний масив цілих чисел (матриця NхM) ініціалізований нулями.

Вивести матрицю у консоль та запропонувати користувачу такі операції над нею:

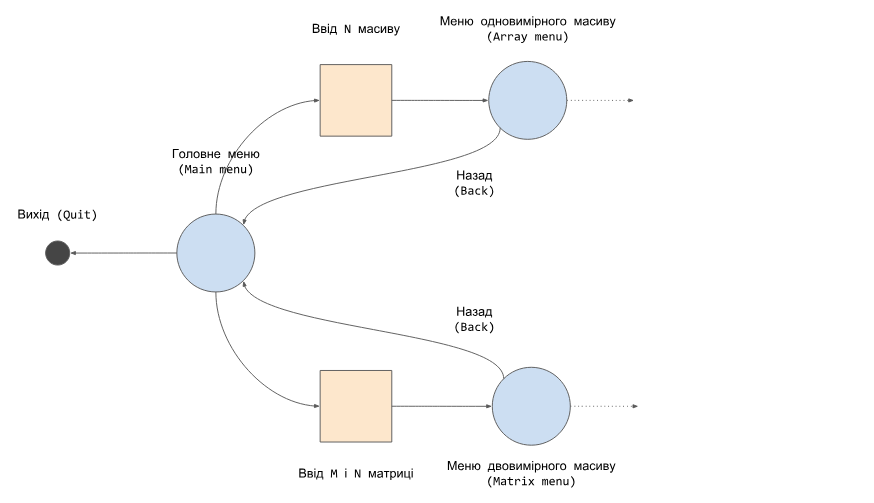
1. Заповнити масив випадковими числами від L до H.
2. Обнулити всі елементи масиву.
3. Знайти мінімальний елемент та його індекси (i та j).
4. Знайти суму елементів рядка за заданим індексом.
5. Поміняти місцями максимальний і мінімальний елементи масиву.
6. Змінити значення елементу за вказаними індексами 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.

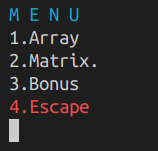


**Тексти коду програм**

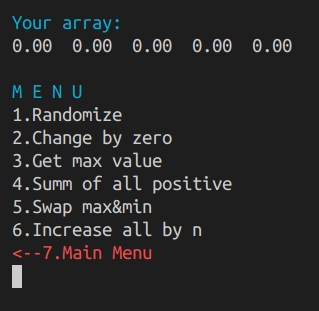
|  |
| --- |
| **main.c** |
| // Компілювати за допомогою:  // gcc main.c -lprogbase -lm  #include <stdio.h> // Для друку в термінал  #include <math.h> // Для математичних функцій  #include <stdlib.h> // Деякі додаткові функції  #include <progbase/console.h> // Спрощений ввід і вивід даних у консоль  #include <time.h>  int getColor(char colorCode); // place before int main() {}  void putsblue(char str[50]);  void putsred(char str[50]);  int main()  {  // Початок програми  Console\_clear();  Console\_reset();  srand(time(0));  int kMain = 0;  do  {  putsblue("M E N U");  printf("1.Array\n");  printf("2.Matrix.\n");  printf("3.Bonus\n");  putsred("4.Escape");  scanf("%i", &kMain);  switch (kMain)  {  //CASE1\_\_ARRAY  case 1:  {  int nArray = 0; //ARRAY SIZE  do  {  Console\_clear();  putsblue("Enter N :");  scanf("%i", &nArray);  } while (nArray < 1);  Console\_clear();  //ARRAY INIZIALIZATION  float array[nArray];  for (int i = 0; i < nArray; i++)  {  array[i] = 0;  }  //MENU 1  int k1 = 0;  do  {  // Console\_clear();  putsblue("\nYour array:");  for (int i = 0; i < nArray; i++)  {  printf("%.2f ", array[i]);  }  puts("\n");  putsblue("M E N U");  puts("1.Randomize");  puts("2.Change by zero");  puts("3.Get max value");  puts("4.Summ of all positive");  puts("5.Swap max&min");  puts("6.Increase all by n");  putsred("<--7.Main Menu");  scanf("%i", &k1);  switch (k1)  {  //RANDOMIZE  case 1:  {  Console\_clear();  int rmax = 0;  int rmin = 0;  do  {  putsblue("Min rand value is:");  scanf("%i", &rmin);  putsblue("Max rand value is:");  scanf("%i", &rmax);  if (rmin > rmax)  {  putsred("Error, min > max!");  }  rmax \*= 100;  rmin \*= 100;  } while (rmin > rmax);  for (int i = 0; i < nArray; i++)  {  // float scale = rand() / (float)rmax;  array[i] = (rand() % (rmax - rmin + 1) + rmin) / 100.0;  }  Console\_clear();  Console\_setCursorAttribute(FG\_CYAN);  printf("Elements between %i and %i\n", rmin / 100, rmax / 100);  Console\_reset();  }  break;  case 2:  {  for (int i = 0; i < nArray; i++)  {  array[i] = 0;  }  Console\_clear();  putsblue("All elements were set to zero");  }  break;  case 3:  {  Console\_clear();  float arrmax = array[0];  int arrmaxindex = 0;  for (int i = 0; i < nArray; i++)  {  if (arrmax < array[i])  {  arrmax = array[i];  arrmaxindex = i;  }  }  Console\_setCursorAttribute(FG\_CYAN);  printf("Element # %i is maximal and equals %.2f\n", arrmaxindex + 1, arrmax);  Console\_reset();  }  break;  case 4:  {  float positiveSumm = 0;  for (int i = 0; i < nArray; i++)  {  if (array[i] > 0)  {  positiveSumm += array[i];  }  }  Console\_clear();  Console\_setCursorAttribute(FG\_CYAN);  printf("The summ of positive elements equals %.2f\n", positiveSumm);  Console\_reset();  }  break;  case 5:  {  Console\_clear();  float arrmax = array[0];  int arrmaxindex = 0;  for (int i = 0; i < nArray; i++)  {  if (arrmax < array[i])  {  arrmax = array[i];  arrmaxindex = i;  }  }  float arrmin = array[0];  int arrminindex = 0;  for (int i = 0; i < nArray; i++)  {  if (arrmin > array[i])  {  arrmin = array[i];  arrminindex = i;  }  }  array[arrminindex] = arrmax;  array[arrmaxindex] = arrmin;  putsblue("Positions of max&min were swaped!");  }  break;  case 6:  {  Console\_clear();  float n = 0;  putsblue("All elements will be increased by:");  scanf("%f", &n);  for (int i = 0; i < nArray; i++)  {  array[i] += n;  }  Console\_clear();  Console\_setCursorAttribute(FG\_CYAN);  printf("All the elements were increased by %f", n);  Console\_reset();  }  break;  case 7:  {  Console\_clear();  }  break;  default:  printf("Error!\nEnter valid list number\n");  break;  }  // Console\_clear();  } while (k1 != 7);  }  break;  case 2:  {  int iMatrix = 0;  int jMatrix = 0;  //INIZIALIZING ARRAY  do  {  Console\_clear();  putsblue("Enter I :");  scanf("%i", &iMatrix);  } while (iMatrix < 1);  do  {  putsblue("Enter J :");  scanf("%i", &jMatrix);  } while (jMatrix < 1);  int matrix[iMatrix][jMatrix];  for (int i = 0; i < iMatrix; i++)  {  for (int j = 0; j < jMatrix; j++)  {  matrix[i][j] = 0;  }  }  int k2 = 0;  puts("\n");  Console\_clear();  do  {  for (int i = 0; i < iMatrix; i++)  {  for (int j = 0; j < jMatrix; j++)  {  printf("%5.1i|", matrix[i][j]);  }  puts("");  }  puts("\n\n");  putsblue("Choose your option:");  puts("1.Randomize");  puts("2.Set to zero");  puts("3.Min vaue");  puts("4.Summ in the choosen line");  puts("5.Swap max&min");  puts("6.Cnage choosen element");  putsred("<--7.Main Menu");  scanf("%i", &k2);  switch (k2)  {  case 1:  {  Console\_clear();  int rmax = 0;  int rmin = 0;  do  {  putsblue("Min rand value is:");  scanf("%i", &rmin);  putsblue("Max rand value is:");  scanf("%i", &rmax);  if (rmin > rmax)  {  putsred("Error, min > max!");  }  } while (rmin > rmax);  for (int i = 0; i < iMatrix; i++)  {  for (int j = 0; j < jMatrix; j++)  {  matrix[i][j] = rand() % (rmax - rmin + 1) + rmin;  }  }  Console\_clear();  putsblue("All elements have random values");  }  break;  case 2:  {  for (int i = 0; i < iMatrix; i++)  {  for (int j = 0; j < jMatrix; j++)  {  matrix[i][j] = 0;  }  }  Console\_clear();  putsblue("All elements were set to zero");  }  break;  case 3:  {  Console\_clear();  int matrixmin = matrix[0][0];  int matrixminindexi = 0;  int matrixminindexj = 0;  for (int i = 0; i < iMatrix; i++)  {  for (int j = 0; j < jMatrix; j++)  {  if (matrixmin > matrix[i][j])  {  matrixmin = matrix[i][j];  matrixminindexi = i;  matrixminindexj = j;  }  }  }  Console\_setCursorAttribute(FG\_CYAN);  printf("Element row # %i column # %i is maximal and equals %i\n", matrixminindexi + 1, matrixminindexj + 1, matrixmin);  Console\_reset();  }  break;  case 4:  {  Console\_clear();  for (int i = 0; i < iMatrix; i++)  {  for (int j = 0; j < jMatrix; j++)  {  printf("%8.1i|", matrix[i][j]);  }  puts("");  }  puts("");  int n4 = 0;  int res = 0;  putsblue("Choose matrix line:");  scanf("%i", &n4);  Console\_clear();  for (int i = 0; i < jMatrix; i++)  {  res += matrix[n4 - 1][i];  }  Console\_setCursorAttribute(FG\_CYAN);  printf("Summ in the line %i equals %i\n", n4, res);  Console\_reset();  }  break;  case 5:  {  Console\_clear();  int matrixmax = matrix[0][0];  for (int i = 0; i < iMatrix; i++)  {  for (int j = 0; j < jMatrix; j++)  {  if (matrixmax < matrix[i][j])  {  matrixmax = matrix[i][j];  }  }  }  int matrixmin = matrix[0][0];  for (int i = 0; i < iMatrix; i++)  {  for (int j = 0; j < jMatrix; j++)  {  if (matrixmin > matrix[i][j])  {  matrixmin = matrix[i][j];  }  }  }  int temp = matrixmax;  for (int i = 0; i < iMatrix; i++)  {  for (int j = 0; j < jMatrix; j++)  {  if (matrix[i][j] == matrixmax)  {  matrix[i][j] = matrixmin;  }  else if (matrixmin == matrix[i][j])  {  matrix[i][j] = temp;  }  }  }  putsblue("Swaped max and min values!");  }  break;  case 6:  {  Console\_clear();  for (int i = 0; i < iMatrix; i++)  {  for (int j = 0; j < jMatrix; j++)  {  printf("%5.1i|", matrix[i][j]);  }  puts("");  }  puts("");  int i6 = 0;  int j6 = 0;  putsblue("Enter line number");  scanf("%i", &i6);  putsblue("Enter column number");  scanf("%i", &j6);  putsblue("Enter new value");  scanf("%i", &matrix[i6 - 1][j6 - 1]);  Console\_clear();  }  break;  case 7:  {  Console\_clear();  }  break;  default:  {  Console\_clear();  putsred("ERROR, choose valid menu option!");  }  break;  }  } while (k2 != 7);  }  break;  case 3:  {  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, 0x6, 0x6, 0x6, 0x6, 0x6, 0x6, 0x6, 0x6, 0x6, 0x6},  {0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x6, 0x6, 0x6, 0x6, 0x6},  {0xF, 0xF, 0xF, 0xF, 0x0, 0x3, 0x3, 0x3, 0x0, 0x0, 0x0, 0x0, 0x0, 0x3, 0x3, 0x3, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x6, 0x6, 0x6, 0x6},  {0xF, 0xF, 0xF, 0xF, 0x0, 0x3, 0x3, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x3, 0x3, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x6, 0xF, 0xF, 0xF, 0x6},  {0xF, 0xF, 0xF, 0xF, 0x0, 0x3, 0x0, 0x0, 0xF, 0xF, 0x7, 0x7, 0x7, 0x0, 0x0, 0x3, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0x6, 0xF, 0xF, 0xF, 0xF, 0x6},  {0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0xF, 0x0, 0xF, 0x0, 0x7, 0x0, 0x7, 0x0, 0xF, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0x6, 0xF, 0xF, 0xF, 0xF, 0xF, 0x6},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0xF, 0xF, 0x7, 0x7, 0x7, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x6, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x6},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0xF, 0xF, 0x0, 0x0, 0x0, 0xF, 0xF, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0xF, 0xF, 0x0, 0x0, 0x0, 0xF, 0xF, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0xF, 0xF, 0x0, 0xF, 0xF, 0x0, 0xF, 0x0, 0xF, 0xF, 0xF, 0x0, 0x1, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0x2, 0x0, 0x0, 0xF, 0xF, 0x3, 0x0, 0xF, 0xF, 0xF, 0x0, 0x1, 0x0, 0xF, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x2, 0x0, 0xF, 0xF, 0xF, 0x3, 0x3, 0x0, 0xF, 0xF, 0xF, 0x0, 0xF, 0x0, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x2, 0x0, 0xF, 0xF, 0xF, 0x3, 0x3, 0x3, 0x0, 0xF, 0xF, 0x0, 0xF, 0x0, 0xF, 0xF, 0xF, 0xF},  {0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0xF, 0x0, 0x0, 0xF, 0x0, 0xF, 0x0, 0x3, 0x3, 0x3, 0x0, 0x0, 0xF, 0xF, 0x0, 0xF, 0xF, 0xF, 0xF},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1},  {0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1}};  char keaster = 0;  do  {  Console\_clear();  char color = 0;  for (int i = 0; i < 28; i++)  {  for (int j = 0; j < 28; j++)  {  color = getColor(image[i][j]);  Console\_setCursorAttribute(color);  printf(" ");  }  puts("");  }  Console\_reset();  Console\_setCursorPosition(30, 2);  putsred("Enter to exit");  // scanf("%i",&keaster);  keaster = Console\_getChar();  } while (keaster != '\n');  Console\_clear();  Console\_reset();  puts("");  }  break;  case 4:  break;  default:  {  Console\_clear();  putsred("Error, wrong value");  printf("\n");  }  break;  }  } while (kMain != 4);  return 0;  }  int getColor(char colorCode)  {  // colors encoding table (hex code -> console color)  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; // we have found our color  }  }  return 0; // it's an error  }  void putsblue(char str[50])  {  Console\_setCursorAttribute(FG\_CYAN);  puts(str);  Console\_reset();  }  void putsred(char str[50])  {  Console\_setCursorAttribute(FG\_INTENSITY\_RED);  puts(str);  Console\_reset();  } |

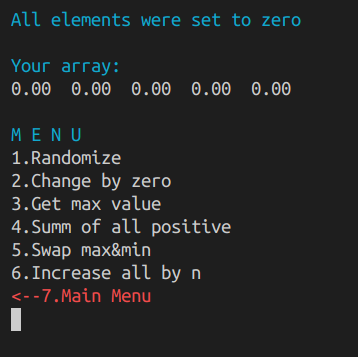
**Приклади результатів**

Головне меню:

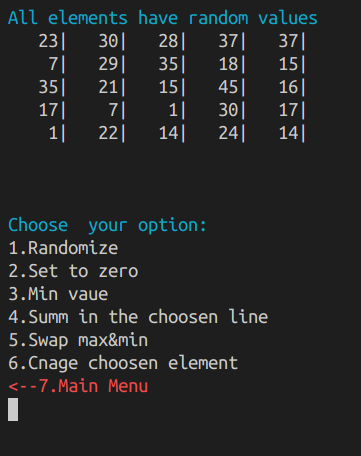


Підменю масиву:

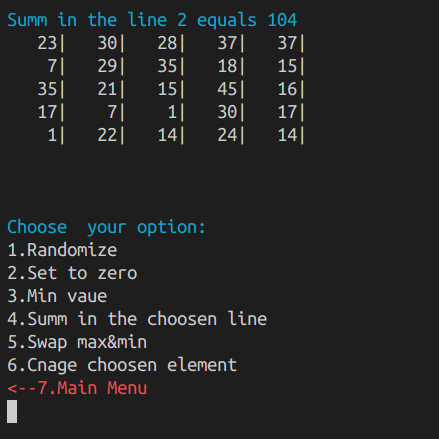


Приклад роботи функцій підменю масивів:

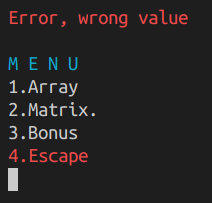
Підменю матриці:



Робота функції підменю матриці:



Робота будь-якої перевірки:



**Висновки**

Виконуючи дану лабораторну роботу, мені довелося ознайомитися з використанням масивів у мові програмування С, а також бібліотекою libprogbase.

Реалізація меню відбулася за допомогою вкладених циклів, а також оператора розгалуження (оператора повного вибору - switch) і функцій для роботи з консоллю бібліотеки libprogbase, наприклад за для зміни кольору шрифту.

Більшість операцій з масивами виконувалися за допомогою різних

циклів, а також стандартних функцій та мови С. На кожному кроці програма перевіряє введені користувачем дані на коректність та може вивести повідомлення про помилку.

Виконуючи бонусне завдання, я зумів познайомитися з принципами кодування інформації у мові С, а саме кодуванням системи кольорів, та відображенням закодованого рисунка.

Компіляція всього коду відбувалася за допомогою компілятора gcc.