

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 №** 2

з дисципліни “ Основи програмування ”

Тема “Робота з одномірними та багатомірними масивами”

|  |  |  |
| --- | --- | --- |
| Виконав  студент I курсу  групи КП-62  Лук’янець Михайло Олександрович  (*прізвище, ім’я, по батькові*)  Варіант №15 |  | Перевірив  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 2016 р.  викладач  Гадиняк Руслан Анатолійович  (*прізвище, ім’я, по батькові*) |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Штрафні бали:   |  |  | | --- | --- | | **Термін здачі** | **Оформлення звіту** | |  |  | | Нараховані бали:   |  |  |  | | --- | --- | --- | | **Корект. програм (2 бала)** | **Відп. на теор. питання (1 бал)** | **Відп. на прогр. питання (2 бала)** | |  |  |  | | Сумарний бал:   |  | | --- | |  | |

Київ 2016

**Мета роботи**

Навчитися працювати зі статичними масивами різних типів даних мови програмування С.

Застосувати на практиці різні види циклічних конструкцій при роботі з одномірними та багатомірними масивами даних.

Вдосконалити вміння роботи з рядками у мові С.

Навчитися оформлювати консольну програму для зручності роботи користувача.

**Постановка завдання**

### **Завдання №1. Одномірний масив**

У цьому стані у користувача є одномірний масив цілих чисел (розміром 10 елементів) ініціалізований нулями.

Доступні операції над масивом:

1. Заповнити масив випадковими числами від -100 до 100.
2. Обнулити всі елементи масиву.
3. Знайти мінімальний елемент та його індекс.
4. Знайти суму елементів масиву.
5. Вивести добуток від'ємних елементів масиву.
6. Знайти перший унікальний символ.
7. Поміняти місцями максимальний і мінімальний елементи масиву.
8. Збільшити всі елементи масиву на введене число.

### **Завдання №2. Два одномірні масиви**

У цьому стані у користувача є 2 одномірних масива дробових чисел (розміром 10) ініціалізованих нулями.

Доступні операції над масивами:

1. Заповнити масиви випадковими дробовими числами від -20.0 до 20.0.
2. Обнулити всі елементи масивів.
3. Знайти значення нового масиву чисел, елементи якого будуть відношенням відповідних елементів першого і другого масиву.
4. Вивести номер масиву, у якому мінімальна сума елементів.
5. Поміняти місцями мінімальний елемент першого масиву і максимальний елемент другого масиву.

### **Завдання №3. Двомірний масив**

У цьому стані у користувача є двомірний масив цілих чисел (розміром 8х8) ініціалізований нулями.

Доступні операції над матрицею:

1. Заповнити масив випадковими числами від -10 до 10.
2. Обнулити всі елементи масиву.
3. Знайти максимальний елемент та його індекси.
4. Знайти суму побічної діагоналі масиву.
5. Знайти суму стовпця за заданим індексом.
6. Поміняти місцями максимальний і мінімальний елементи масиву.
7. Змінити значення елементу за вказаними індексами на задане.

### **Завдання №4. Обробка рядків мови С**

У цьому стані у користувача є буфер (розміром 80 символів) для зберігання строки. У початковому стані буфер повинен містити будь-яку видиму строку.

Доступні операції над рядком:

1. Заповнити строку вводом із консолі.
2. Очистити строку.
3. Вивести довжину строки.
4. Вивести підстроку із заданої позиції і заданої довжини.
5. Вивести список підстрок, розділених заданим символом.
6. Вивести найкоротше слово (слова - непуста послідовність буквенних символів).
7. Знайти та вивести всі дробові числа, що містяться у строці.
8. Знайти та вивести добуток всіх цілих чисел, що містяться у строці.

**Тексти коду програм**

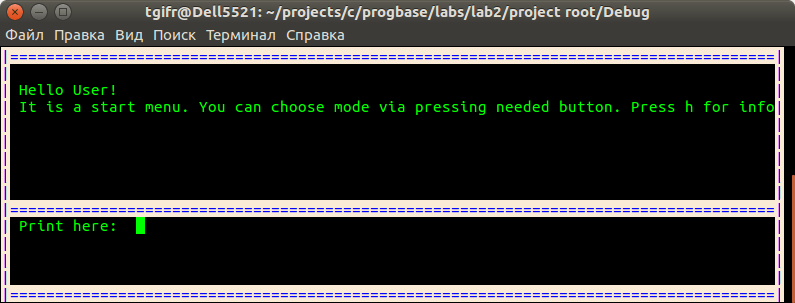
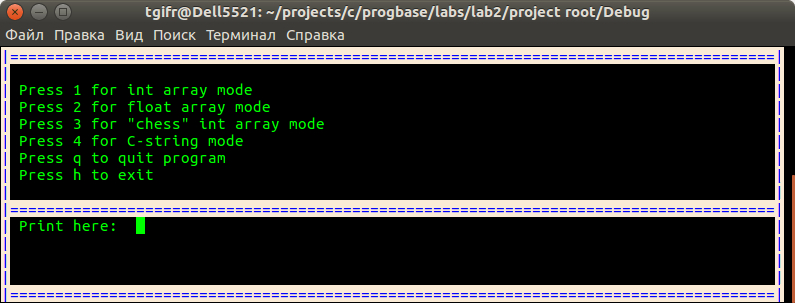
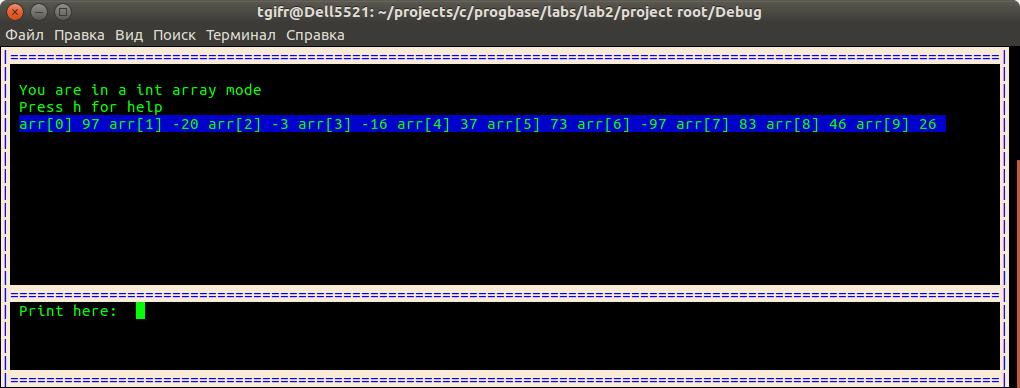
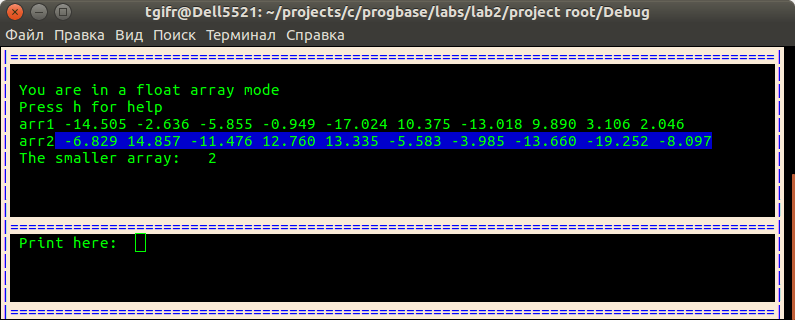
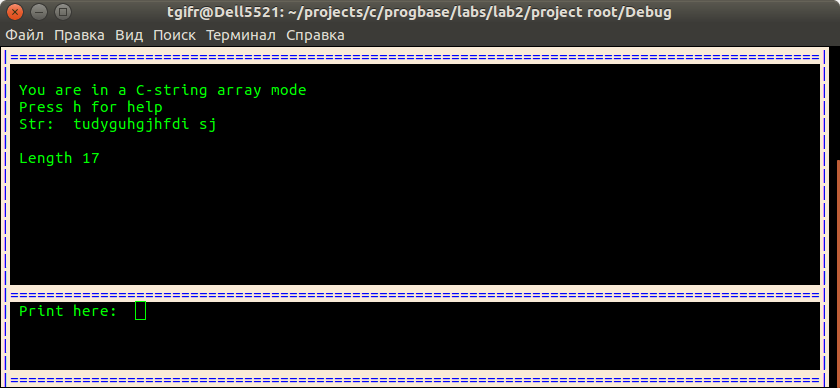
**Завдання 1.**

|  |
| --- |
| math.c |
| #include <stdio.h>  #include <stdlib.h>  #include <progbase.h>  #include <pbconsole.h>  #include <time.h>  #include <string.h>  **void** redraw(**int** cols, **int** rows);  **void** clearAnim();  **void** printText(**const char** \*TEXT);  **void** printInfoText(**const char** \*HELP);  **void** printIntArray(**int** arr[10], **int** bool[10], **const char** \*TEXT);  **void** printFloatArray(**float** arr1[10], **float** arr2[10], **int** bool1[10], **int** bool2[10]);  **void** printInt2Array(**int** buf[8][8], **int** bool[8][8]);  **void** printStrArray(**char** \*buf, **int** bool[80]);  **void** modeIntArray();  **void** modeFloatArray();  **void** modeInt2Array();  **void** modeStrArray();  **void** conDefPos();  **void** cleanInp();  **void** cleanCanvas();  **struct** conShift{  **short** cols;  **short** rows;  };  **const struct** consize DEF\_SIZE = {87, 25};  **const int** MILLIS\_WAIT = 30;  **int** main()  {  **char** key = '\0';  **const struct** conShift MAIN\_SHIFT = {0, -10};  **const char** \*START\_TEXT = "Hello User!\n" \  "It is a start menu. You can choose mode via pressing needed button. Press h for info\0";  **const char** \*START\_INFO = "Press 1 for int array mode\n"  "Press 2 for float array mode\n"  "Press 3 for \"chess\" int array mode\n"  "Press 4 for C-string mode\n"  "Press q to quit program\n"  "Press h to exit\n\0";  **const char** \*exitText = " \_\_\_\_\_ \_\_\_\_ \_\_\_\_ \_\_\_\_\_ \_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \n"  " / \_\_\_\_|/ \_\_ \\ / \_\_ \\| \_\_ \\| \_ \\ \\ / / \_\_\_\_|\n"  " | | \_\_| | | | | | | | | | |\_) \\ \\\_/ /| |\_\_ \n"  " | | |\_ | | | | | | | | | | \_ < \\ / | \_\_| \n"  " | |\_\_| | |\_\_| | |\_\_| | |\_\_| | |\_) | | | | |\_\_\_\_ \n"  " \\\_\_\_\_\_|\\\_\_\_\_/ \\\_\_\_\_/|\_\_\_\_\_/|\_\_\_\_/ |\_| |\_\_\_\_\_\_|\n";  redraw(MAIN\_SHIFT.cols , MAIN\_SHIFT.rows );  printText(START\_TEXT);  **while**(key != 'q'){  key = conGetChar();  **switch** (key) {  **case** '1':  modeIntArray();  redraw(MAIN\_SHIFT.cols , MAIN\_SHIFT.rows );  printText(START\_TEXT);  **break**;  **case** '2':  modeFloatArray();  redraw(MAIN\_SHIFT.cols , MAIN\_SHIFT.rows );  printText(START\_TEXT);  **break**;  **case** '3':  modeInt2Array();  redraw(MAIN\_SHIFT.cols , MAIN\_SHIFT.rows );  printText(START\_TEXT);  **break**;  **case** '4':  modeStrArray();  redraw(MAIN\_SHIFT.cols , MAIN\_SHIFT.rows );  printText(START\_TEXT);  **break**;  **case** 'h':  cleanCanvas();  printInfoText(START\_INFO);  cleanCanvas();  printText(START\_TEXT);  **break**;  **default**: ;  }  }  clearAnim();  conMove(1,1);  printf("%s", exitText);  sleepMillis(MILLIS\_WAIT \* 100);  conClear();  **return** 0;  }  **void** redraw(**int** cols, **int** rows){  **int** i = 0;  conClear();  conResize(DEF\_SIZE.rows + (**unsigned short**)rows, DEF\_SIZE.cols + (**unsigned short**)cols);  **const struct** consize actSize = { DEF\_SIZE.cols + (**unsigned short**)cols, DEF\_SIZE.rows + (**unsigned short**)rows};  conSetAttr(*BG\_WHITE*);  conSetAttr(*FG\_INTENSITY\_BLUE*);  **for**(i = 1; i <= actSize.cols; i ++){  conMove(1, i );  printf("=");  conMove(actSize.rows - 5, i);  printf("=");  conMove(actSize.rows, i );  printf("=");  fflush(stdout);  sleepMillis(MILLIS\_WAIT/3);  }  **for**(i = 1; i <= actSize.rows; i ++){  conMove(i , 1);  printf("|");  conMove(i , actSize.cols);  printf("|");  fflush(stdout);  sleepMillis(MILLIS\_WAIT);  }  conReset();  conMove( actSize.rows - 4, 3);  printf("Print here: ");  }  **void** clearAnim(){  **int** i = 0;  **int** j = 0;  **const struct** consize size = conGetSize();  **for**(i = 1; i <= size.cols; i ++){  conSetAttr(*BG\_INTENSITY\_BLACK*);  **for**(j = 1; j <= size.rows; j ++){  conMove(j, i);  printf(" ");  conMove(j, i + 1);  printf(" ");  }  conSetAttr(*BG\_DEFAULT*);  **if** (i != 1) {  **for**(j = 1; j <= size.rows; j ++){  conMove(j, i - 1);  printf(" ");  }  }  **if** (i == size.cols) {  **for**(j = 1; j <= size.rows; j ++){  conMove(j, i);  printf(" ");  }  }  fflush(stdout);  sleepMillis(MILLIS\_WAIT);  }  conReset();  conDefPos();  }  **void** printText(**const char** \*TEXT){  **int** i = 0, yShift = 0;  conMove(3, 3);  **for**(i = 0; i < strlen(TEXT); i ++){  **if**(TEXT[i] == '\n'){  yShift++;  conMove(3 + yShift, 3);  } **else** printf("%c", TEXT[i]);  }  conDefPos();  }  **void** printInfoText(**const char** \*HELP){  **char** key = '\0';  cleanCanvas();  printText(HELP);  **while** (key != 'h'){  key = conGetChar();  }  cleanInp();  }  **void** conDefPos(){  **struct** consize actSize = conGetSize();  conMove(actSize.rows - 4, 3 + **sizeof**("Print here: "));  }  **void** cleanInp(){  **int** i = 0;  **struct** consize actSize = conGetSize();  **for**(i = **sizeof**("Print here: ") + 3; i < actSize.cols - 2; i++){  conMove(actSize.rows - 4, i);  printf(" ");  conMove(actSize.rows - 3, i - **sizeof**("Print here: "));  printf(" ");  }  conDefPos();  }  **void** cleanCanvas(){  **int** i = 0;  **int** j = 0;  conReset();  **struct** consize actSize = conGetSize();  **for**(j = 2; j < actSize.rows - 5; j ++){  **for**(i = 2; i < actSize.cols; i++){  conMove(j, i);  printf(" ");  }  }  conDefPos();  }  **void** modeIntArray(){  **const int** SIZE = 10;  **const struct** conShift INT\_ARR\_SHIFT = { 25, -5};  **int** min, max, index, product , sum;  **char** key = '\0';  **int** bool[SIZE];  **const char** \*ARRAY\_TEXT = "You are in a int array mode\n"  "Press h for help";  **const char** \*ARRAY\_INFO = "Press 1 to fill with random numbers\n"  "Press 2 to make all elem 0\n"  "Press 3 to find minimal and it's index\n"  "Press 4 to find sum\n"  "Press 5 to find product of elem, that < 0\n"  "Press 6 to find first unique number\n"  "Press 7 to change max and min\n"  "Press 8 to increase all\n"  "Press q to exit\n"  "Press h to close help\n";  **int** arr[SIZE];  clearAnim();  redraw( INT\_ARR\_SHIFT.cols , INT\_ARR\_SHIFT.rows);  srand((**unsigned short**)time(0));  **for**( **int** i = 0; i < SIZE; i ++){  arr[i] = 0;  bool[i] = 0;  }  printIntArray(arr, bool, ARRAY\_TEXT);  **while**(key != 'q'){  key = conGetChar();  **switch** (key) {  **case** '1':  **for**( **int** i = 0; i < SIZE; i ++){  arr[i] = rand() % 201 - 100;  bool[i] = 1;  }  printIntArray(arr, bool, ARRAY\_TEXT);  **break**;  **case** '2':  **for**( **int** i = 0; i < SIZE; i ++){  arr[i] = 0;  bool[i] = 1;  }  printIntArray(arr, bool, ARRAY\_TEXT);  **break**;  **case** '3':  min = arr[0];  index = 0;  **for**( **int** i = 1; i < SIZE; i ++){  **if** (arr[i] < min){  min = arr[i];  index = i;  }  }  bool[index] = 1;  printIntArray(arr, bool, ARRAY\_TEXT);  conMove(7, 3);  printf("Minimal: %i It's index: %i", min, index);  conDefPos();  **break**;  **case** '4':  sum = 0;  **for**(**int** i = 0; i < SIZE; i ++){  sum += arr[i];  }  printIntArray(arr, bool, ARRAY\_TEXT);  conMove(7, 3);  printf("Sum: %i", sum);  conDefPos();  **break**;  **case** '5':  product = 1;  **int** flag = 0;  **for**(**int** i = 0; i < SIZE; i ++){  **if** (arr[i] < 0) {  product \*= arr[i];  bool[i] = 1;  flag = 1;  }  }  printIntArray(arr, bool, ARRAY\_TEXT);  conMove(7, 3);  **if**( flag == 0) {  printf("There are no such elements");  **break**;  }  printf("Product: %i", product);  conDefPos();  **break**;  **case** '6':  index;  flag = 0;  **for**(index = 0; index < SIZE; index ++){  **for** ( **int** i = index + 1; i < SIZE; ++i) {  **if**( arr[i] == arr[index]){  flag = 1;  }  }  **if**(!flag){  bool[index] = 1;  **break**;  }  flag = 0;  }  printIntArray(arr, bool, ARRAY\_TEXT);  conMove(7, 3);  **if**(flag == 0 && arr[0] == 0){  printf("THAT'S A TRUE RANDOM! ABSOLUTELY FANTASTIC! Or there are just all zeroes...");  **break**;  }  printf("First unique number is: %i", arr[index]);  conDefPos();  **break**;  **case** '7':  min = 0;  **for**(**int** i = 1; i < SIZE; i ++){  **if** (arr[i] < arr[min]){  min = i;  }  }  max = 0;  **for**(**int** i = 1; i < SIZE; i ++){  **if** (arr[i] > arr[max]){  max = i;  }  }  arr[min] += arr[max];  arr[max] = arr[min] - arr[max];  arr[min] = arr[min] - arr[max];  bool[min] = 1;  bool[max] = 1;  printIntArray(arr, bool, ARRAY\_TEXT);  conMove(7, 3);  printf("Min and max are changed");  conDefPos();  **break**;  **case** '8':  conMove(7, 3);  printf("Enter an integer, please");  conDefPos();  **int** toAdd = getInt();  cleanInp();  **for** (**int** i = 0; i < SIZE; ++i) {  arr[i] += toAdd;  }  printIntArray(arr, bool, ARRAY\_TEXT);  conMove(7, 3);  printf("All increased by %i", toAdd);  conDefPos();  **break**;  **case** 'h':  printInfoText(ARRAY\_INFO);  printIntArray(arr, bool, ARRAY\_TEXT);  **break**;  **default**:  **break**;  }  }  clearAnim();  }  **void** modeFloatArray(){  **const int** SIZE = 10;  **const struct** conShift FLOAT\_ARR\_SHIFT = { 0, -9};  **int** i = 0, min, max, sum1 = 0, sum2 = 0;  **char** key = '\0';  **int** bool1[SIZE], bool2[SIZE];  **const char** \*F\_ARRAY\_TEXT = "You are in a float array mode\n"  "Press h for help";  **const char** \*F\_ARRAY\_INFO = "Press 1 to fill with random numbers\n"  "Press 2 to make all elem 0\n"  "Press 3 to find the arr1/arr2 array\n"  "Press 4 to find the smallest sum\n"  "Press 5 to change max from 2 and min from 1\n"  "Press q to exit\n"  "Press h to close help\n";  **float** arr1[SIZE], arr2[SIZE];  clearAnim();  redraw(FLOAT\_ARR\_SHIFT.cols , FLOAT\_ARR\_SHIFT.rows);  srand((**unsigned short**)time(0));  **for**(i = 0; i < SIZE; i ++){  arr1[i] = 0;  bool1[i] = 0;  arr2[i] = 0;  bool2[i] = 0;  }  printText(F\_ARRAY\_TEXT);  printFloatArray(arr1, arr2, bool1, bool2);  **while**(key != 'q'){  key = conGetChar();  **switch** (key) {  **case** '1':  **for**(i = 0; i < SIZE; i ++){  arr1[i] = ((**float**)rand()/(**float**)(RAND\_MAX)) \* 40.F - 20.F;  bool1[i] = 1;  arr2[i] = ((**float**)rand()/(**float**)(RAND\_MAX)) \* 40.F - 20.F;  bool2[i] = 1;  }  cleanCanvas();  printText(F\_ARRAY\_TEXT);  printFloatArray(arr1, arr2, bool1, bool2);  **break**;  **case** '2':  **for**(i = 0; i < SIZE; i ++){  arr1[i] = 0;  bool1[i] = 1;  arr2[i] = 0;  bool2[i] = 1;  }  cleanCanvas();  printText(F\_ARRAY\_TEXT);  printFloatArray(arr1, arr2, bool1, bool2);  **break**;  **case** '3':  cleanCanvas();  conSetAttr(*BG\_BLUE*);  conMove( 9, 3);  **for**(i = 0; i < SIZE; i ++){  **if** (arr2[i] == 0){  printf(" NOPE");  } **else** {  printf(" %.3f" , (arr1[i] / arr2[i]));  }  }  conReset();  printText(F\_ARRAY\_TEXT);  printFloatArray(arr1, arr2, bool1, bool2);  conMove(7, 3);  printf("The arr1/arr2 array:");  conDefPos();  **break**;  **case** '4':  sum1 = 0;  sum2 = 0;  **for**(i = 0; i < SIZE; i ++){  sum1 += arr1[i];  sum2 += arr2[i];  }  cleanCanvas();  printText(F\_ARRAY\_TEXT);  conMove(7, 3);  printf("The smaller array: ");  conMove(7, 3 + **sizeof**("The smaller array: "));  **if**( sum1 == sum2){  printf(" They are same");  }**else if**(sum1 < sum2){  printf(" 1");  **for** ( i = 0; i < SIZE; ++i) {  bool1[i] = 1;  }  } **else**{  printf(" 2");  **for** ( i = 0; i < SIZE; ++i) {  bool2[i] = 1;  }  }  printFloatArray(arr1, arr2, bool1, bool2);  conDefPos();  **break**;  **case** '5':  min = 0;  max = 0;  **for**(i = 0; i < SIZE; i ++){  **if** (arr1[i] < arr1[min]) {  min = i;  }  **if** (arr2[i] > arr2[max]) {  max = i;  }  }  bool1[min] = 1;  bool2[max] = 1;  arr1[min] += arr2[max];  arr2[max] = arr1[min] - arr2[max];  arr1[min] = arr1[min] - arr2[max];  cleanCanvas();  printText(F\_ARRAY\_TEXT);  printFloatArray(arr1, arr2, bool1, bool2);  conMove(7, 3);  printf("Max and min changed");  conDefPos();  **break**;  **case** 'h':  printInfoText(F\_ARRAY\_INFO);  cleanCanvas();  printText(F\_ARRAY\_TEXT);  printFloatArray(arr1, arr2, bool1, bool2);  **break**;  **default**:  **break**;  }  }  clearAnim();  }  **void** modeInt2Array(){  **const int** SIZE = 8;  **const struct** conShift INT2\_ARR\_SHIFT = { 0, 0};  **int** i = 0, min, max, index = 0, sum = 0, flag = 0;  **char** key = '\0';  **int** indexArr[4];  **int** bool[SIZE][8];  **const char** \*ARRAY2\_TEXT = "You are in a 8x8 int array mode\n"  "Press h for help";  **const char** \*ARRAY2\_INFO = "Press 1 to fill with random numbers\n"  "Press 2 to make all 0\n"  "Press 3 to find max elem and it's index\n"  "Press 4 to find sum of diag\n"  "Press 5 to find sum of col\n"  "Press 6 to change max and min\n"  "Press 7 to change needed elem\n"  "Press q to exit\n"  "Press h to close help\n";  **int** arr[SIZE][8];  clearAnim();  redraw(INT2\_ARR\_SHIFT.cols , INT2\_ARR\_SHIFT.rows);  srand((**unsigned short**)time(0));  **for**(i = 0; i < SIZE\*SIZE; i ++){  arr[0][i] = 0;  bool[0][i] = 0;  }  printText(ARRAY2\_TEXT);  printInt2Array(arr, bool);  **while**(key != 'q'){  key = conGetChar();  **switch** (key) {  **case** '1':  **for**(i = 0; i < SIZE\*SIZE; i ++){  arr[0][i] = rand() % 21 - 10;  bool[0][i] = 1;  }  cleanCanvas();  printText(ARRAY2\_TEXT);  printInt2Array(arr, bool);  **break**;  **case** '2':  **for**(i = 0; i < SIZE\*SIZE; i ++){  arr[0][i] = 0;  bool[0][i] = 1;  }  cleanCanvas();  printText(ARRAY2\_TEXT);  printInt2Array(arr, bool);  **break**;  **case** '3':  max = arr[0][0];  **for**(i = 1; i < SIZE\*SIZE; i ++){  **if** (arr[0][i] > max){  max = arr[0][i];  }  }  cleanCanvas();  conMove(15, 3 + **sizeof**("Maximal: %i It's index: "));  **for** (i = 0; i < SIZE\*SIZE; i ++) {  **if** (arr[0][i] == max) {  bool[0][i] = 1;  printf(" [ [%i] [%i] ] ", i / 8, i % 8);  }  }  printText(ARRAY2\_TEXT);  printInt2Array(arr, bool);  conMove(15, 3);  printf("Maximal: %i It's index: ", max);  conDefPos();  **break**;  **case** '4':  sum = 0;  **for**(i = 0; i < SIZE; i ++){  sum += arr[i][ 8 - 1 - i];  bool[i][ 8 - 1 - i] = 1;  }  cleanCanvas();  printText(ARRAY2\_TEXT);  printInt2Array(arr, bool);  conMove(15, 3);  printf("Sum: %i", sum);  conDefPos();  **break**;  **case** '5':  conMove(15,3);  printf("Enter an integer: ");  conDefPos();  index = getInt();  cleanInp();  **if**(index < 0 || index > 7){  conMove(15, 3);  printf("NOPE. NOT CORRECT NUMBER");  **break**;  }  sum = 0;  **for**(i = 0; i < SIZE; i ++){  sum += arr[i][index];  bool[i][index] = 1;  }  cleanCanvas();  printText(ARRAY2\_TEXT);  printInt2Array(arr, bool);  conMove(15, 3);  printf("Sum of %i col: %i", index, sum);  conDefPos();  **break**;  **case** '6':  min = 0;  **for**(i = 1; i < SIZE\*SIZE; i ++){  **if** (arr[0][i] < arr[0][min]){  min = i;  }  }  max = 0;  **for**(i = 1; i < SIZE\*SIZE; i ++){  **if** (arr[0][i] > arr[0][max]){  max = i;  }  }  arr[0][min] += arr[0][max];  arr[0][max] = arr[0][min] - arr[0][max];  arr[0][min] = arr[0][min] - arr[0][max];  bool[0][min] = 1;  bool[0][max] = 1;  cleanCanvas();  printText(ARRAY2\_TEXT);  printInt2Array(arr, bool);  conMove(15, 3);  printf("Min and max are changed");  conDefPos();  **break**;  **case** '7':  flag = 0;  conMove(15,3);  printf("Enter a first col: ");  conDefPos();  indexArr[0] = getInt();  cleanInp();  conMove(15,3);  printf("Enter a first row: ");  conDefPos();  indexArr[1] = getInt();  cleanInp();  conMove(15,3);  printf("Enter a second col: ");  conDefPos();  indexArr[2] = getInt();  cleanInp();  conMove(15,3);  printf("Enter a second row: ");  conDefPos();  indexArr[3] = getInt();  cleanInp();  **for** ( i = 0; i < 4; ++i) {  **if**(indexArr[i] < 0 || indexArr[i] > SIZE){  conMove(15,3);  printf("NOPE. NOT CORRECT NUMBER");  flag = 1;  }  }  cleanCanvas();  printText(ARRAY2\_TEXT);  **if**(!flag){  arr[indexArr[1]][indexArr[0]] += arr[indexArr[3]][indexArr[2]];  arr[indexArr[3]][indexArr[2]] = arr[indexArr[1]][indexArr[0]] - arr[indexArr[3]][indexArr[2]];  arr[indexArr[1]][indexArr[0]] = arr[indexArr[1]][indexArr[0]] - arr[indexArr[3]][indexArr[2]];  bool[indexArr[1]][indexArr[0]] = 1;  bool[indexArr[3]][indexArr[2]] = 1;  conMove(15, 3);  printf("Numbers changed");  }  printInt2Array(arr, bool);  conDefPos();  **break**;  **case** 'h':  printInfoText(ARRAY2\_INFO);  cleanCanvas();  printText(ARRAY2\_TEXT);  printInt2Array(arr, bool);  **break**;  **default**:  **break**;  }  }  clearAnim();  }  **void** modeStrArray(){  **const int** SIZE = 80;  **const struct** conShift STR\_ARR\_SHIFT = { 5, -5};  **int** i = 0, tmp1 = 0, product , wordStart, length, posStart, posEnd, count, flag;  **double** tmp ;  **char** key = '\0';  **int** bool[SIZE];  **char** divider;  **const char** \*STR\_TEXT = "You are in a C-string array mode\n"  "Press h for help";  **const char** \*STR\_INFO = "Press 1 to fill string\n"  "Press 2 to clean string\n"  "Press 3 to find length\n"  "Press 4 to show part of string\n"  "Press 5 to show part-strings, divided by char\n"  "Press 6 to find the shortest word\n"  "Press 7 to find floats\n"  "Press 8 to find product of integers\n"  "Press q to exit\n"  "Press h to close help\n";  **char** arr[80] = "\0";  **char** corTest[200] = "";  **for** ( i = 0; i < 80; ++i) {  bool[i] = 0;  }  clearAnim();  redraw( STR\_ARR\_SHIFT.cols, STR\_ARR\_SHIFT.rows);  srand((**unsigned short**)time(0));  printText(STR\_TEXT);  printStrArray(arr, bool);  **while**(key != 'q'){  key = conGetChar();  **switch** (key) {  **case** '1':  cleanCanvas();  printText(STR\_TEXT);  conMove(7, 3);  printf("Enter str");  conDefPos();  strcpy(corTest, getString());  cleanInp();  **if**(strlen(corTest) > SIZE - 1){  conMove(7, 3);  printf("ERROR: TOO LONG");  } **else** {  strcpy(arr, corTest);  strcpy(corTest, "");  }  cleanInp();  printText(STR\_TEXT);  printStrArray(arr, bool);  **break**;  **case** '2':  strcpy(arr, "");  cleanCanvas();  conMove(7, 3);  printf("Str cleaned");  printText(STR\_TEXT);  printStrArray(arr, bool);  **break**;  **case** '3':  cleanCanvas();  printText(STR\_TEXT);  printStrArray(arr, bool);  conMove(7, 3);  printf("Length %lu", strlen(arr));  conDefPos();  **break**;  **case** '4':  cleanCanvas();  printText(STR\_TEXT);  printStrArray(arr, bool);  conMove(7,3);  printf("Enter a start position: ");  conDefPos();  wordStart = getInt();  cleanInp();  cleanCanvas();  printText(STR\_TEXT);  printStrArray(arr, bool);  conMove(7,3);  printf("Enter length: ");  conDefPos();  length = getInt();  cleanInp();  cleanCanvas();  printText(STR\_TEXT);  printStrArray(arr, bool);  conMove(7, 3);  **if**(length > strlen(arr) - wordStart){  printf("WRONG DATA");  } **else** {  printf("Your string: ");  **for** (i = 0; i < length; i++) {  printf("%c", arr[i + wordStart]);  bool[i] = 1;  }  }  conDefPos();  **break**;  **case** '5':  cleanCanvas();  printText(STR\_TEXT);  printStrArray(arr, bool);  conMove(7,3);  printf("Enter a divider(char): ");  conDefPos();  divider = conGetChar();  cleanCanvas();  conMove(7, 3);  printf("Your string: ");  count = 0;  **for** (i = 0; i < strlen(arr); i++) {  **if**(arr[i] == divider){  count ++;  conMove(7 + count, 3);  } **else** {  printf("%c", arr[i]);  bool[i] = 1;  }  }  printText(STR\_TEXT);  printStrArray(arr, bool);  conDefPos();  **break**;  **case** '6':  posStart = 0;  posEnd = SIZE;  wordStart = 0;  **for**(i = 0; i < strlen(arr); i ++){  **if**(arr[i] == ' '){  **if**(i - wordStart < posEnd - posStart && i != wordStart) {  posEnd = i;  posStart = wordStart;  }  wordStart = i + 1;  } **else if**( i == strlen(arr) - 1){  **if**(i - wordStart + 1 < posEnd - posStart) {  posEnd = i + 1;  posStart = wordStart;  }  }  }  cleanCanvas();  conMove(7, 3);  printf("Your word: ");  **for** (i = posStart; i < posEnd; i++) {  printf("%c", arr[i]);  bool[i] = 1;  }  printText(STR\_TEXT);  printStrArray(arr, bool);  **break**;  **case** '7':  cleanCanvas();  printText(STR\_TEXT);  conMove(7,3);  **for**(i = 0; i < strlen(arr); i ++){  **if**(arr[i] == ' ' && arr[i-1] != ' '){  tmp = atof(&arr[i]);  **if**(tmp == 0 && arr[i + 1] != '0'){  **continue**;  }  printf(" %.3f", tmp);  } **else if**( i == 0){  tmp = atof(&arr[i]);  **if**(tmp == 0 && arr[i + 1] != '0'){  **continue**;  }  printf(" %.3f", tmp);  }  }  printStrArray(arr, bool);  **break**;  **case** '8':  cleanCanvas();  printText(STR\_TEXT);  conMove(7,3);  flag = 0;  product = 1;  **for**(i = 0; i < strlen(arr); i ++){  **if**(arr[i] == ' ' && arr[i-1] != ' '){  tmp1 = atoi(&arr[i]);  **if**(tmp1 == 0 && arr[i + 1] != '0'){  **continue**;  }  product \*= tmp1;  flag = 1;  } **else if**( i == 0){  tmp1 = atoi(&arr[i]);  **if**(tmp1 == 0 && arr[i + 1] != '0'){  **continue**;  }  product \*= tmp1;  flag = 1;  }  }  **if**(!flag){  printf("NO INTEGERS");  **break**;  }  printText(STR\_TEXT);  printStrArray(arr, bool);  conMove(7, 3);  printf("Product: %i", product);  conDefPos();  **break**;  **case** 'h':  printInfoText(STR\_INFO);  cleanCanvas();  printText(STR\_TEXT);  printStrArray(arr, bool);  **break**;  **default**:  **break**;  }  }  clearAnim();  }  **void** printIntArray(**int** arr[10], **int** bool[10], **const char** \*TEXT){  **int** i = 0;  cleanCanvas();  conMove(5,3);  **for** ( i = 0; i < 10; ++i) {  **if**(bool[i] == 1){  conSetAttr(*BG\_BLUE*);  }  printf("arr[%i] %i ", i, arr[i]);  **if**(bool[i] == 1){  conReset();  bool[i] = 0;  }  }  printText(TEXT);  }  **void** printFloatArray(**float** arr1[10], **float** arr2[10], **int** bool1[10], **int** bool2[10]){  **int** i = 0;  conMove(5,3);  printf("arr1");  **for** ( i = 0; i < 10; ++i) {  **if**(bool1[i] == 1){  conSetAttr(*BG\_BLUE*);  }  printf(" %.3f", arr1[i]);  **if**(bool1[i] == 1){  conReset();  bool1[i] = 0;  }  }  conMove(6,3);  printf("arr2");  **for** ( i = 0; i < 10; ++i) {  **if**(bool2[i] == 1){  conSetAttr(*BG\_BLUE*);  }  printf(" %.3f", arr2[i]);  **if**(bool2[i] == 1){  conReset();  bool2[i] = 0;  }  }  conDefPos();  }  **void** printInt2Array(**int** buf[8][8], **int** bool[8][8]){  **int** i = 0;  **int** j = 0;  **for** ( i = 0; i < 8; ++i) {  conMove(5 + i,3);  **for** (j = 0; j < 8; ++j) {  **if**(bool[i][j] == 1){  conSetAttr(*BG\_BLUE*);  }  printf("%i ", buf[i][j]);  **if**(bool[i][j] == 1){  conReset();  bool[i][j] = 0;  }  }  }  conDefPos();  }  **void** printStrArray(**char** \*buf, **int** bool[80]){  **int** i = 0;  conMove(5, 3);  printf("Str: ");  **for** (i = 0; i < strlen(buf); i++) {  **if**(bool[i] == 1){  conSetAttr(*BG\_BLUE*);  }  printf("%c", buf[i]);  **if**(bool[i] == 1){  conReset();  bool[i] = 0;  }  }  conDefPos();  } |

**Приклади результатів**

**Завдання 1, 2, 3, 4.**

Вікна программи у різних режимах роботи.



**Висновки**

Виконавши дану лабораторну роботу була проведена робота зі статичними масивами різних типів даних мови програмування С.

При роботі з одномірними та багатомірними масивами були застосовані різні види циклічних конструкцій мови С.

Також, за допомогою конструкцій розгалуження, була виконнана обробка рядків мовою С.

Для зручності роботи користувача було створене оформлення программи з розділенням вікна программи на зону відображення та зону вводу.

Додатково, для пришвидшення роботи була використана можливість вводу команд за допомогою натискання однієї клавіші.

Також, для поркащення сприйняття користувачем інформації, була реаліована можливість підсвітки важливих частини оброблюваних массивів та зміна розміру вікна программи.

Також була проведена робота по створенню інтерфейса программи з поркащенним виглядом.