**ПРАКТИЧНЕ ЗАВДАННЯ № 6**

**Завдання № 1:** Напишіть програму, яка здійснювати поворот квадрату 3x3 у різних напрямках. Напрямок повороту має вибирати користувач, тому передбачте для нього таку можливість.

**Код програми:**

**/\* Turns of the square**

**\* This program flips the square matrix of ANY DIMENSION in all directions,**

**\* not only 3 by 3. You only need to resize and fill in the initial matrix \*/**

**#include <stdio.h>**

**#include <ctype.h>**

**#include <stdbool.h>**

**#include <Windows.h>**

**#define BLACK 0**

**#define BLUE 1**

**#define GREEN 2**

**#define RED 4**

**#define WHITE 15**

**#define edge 4 // the size can be changed**

**void rotate\_left(int value, int row, int col, int res\_square[edge][edge]);**

**void rotate\_right(int value, int row, int col, int res\_square[edge][edge]);**

**void rotate\_180(int value, int row, int col, int res\_square[edge][edge]);**

**void flip\_vertical(int value, int row, int col, int res\_square[edge][edge]);**

**void flip\_horizontal(int value, int row, int col, int res\_square[edge][edge]);**

**int main(void) {**

**bool launch = true; // permission to repeat the program**

**char mode;**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**system("color F0"); // white background, black font**

**system("mode 60, 30");**

**const int square[edge][edge] = { {1, 2, 3, 4},**

**{5, 6, 7, 8},**

**{9, 10, 11, 12},**

**{13, 14, 15, 16} };**

**int transposed[edge][edge];**

**while (launch) {**

**printf("Enter the operation of your choice:\n"**

**"A. Rotate left\t\tB. Rotate right\n"**

**"C. Rotate 180\t\tD. Flip vertical\n"**

**"E. Flip horizontal\tQ. Quit\n");**

**SetConsoleTextAttribute(hConsole, (WORD)((WHITE << 4) | BLUE));**

**printf("Your choice: ");**

**SetConsoleTextAttribute(hConsole, (WORD)((WHITE << 4) | BLACK));**

**mode = getchar();**

**fflush(stdin); // clearing the clipboard**

**mode = toupper(mode); // ignore case**

**switch (mode) {**

**case 'A':**

**for (int row = 0; row < edge; row++) {**

**for (int col = 0; col < edge; col++) {**

**rotate\_left(square[row][col], row, col, transposed);**

**}**

**}**

**break;**

**case 'B':**

**for (int row = 0; row < edge; row++) {**

**for (int col = 0; col < edge; col++) {**

**rotate\_right(square[row][col], row, col, transposed);**

**}**

**}**

**break;**

**case 'C':**

**for (int row = 0; row < edge; row++) {**

**for (int col = 0; col < edge; col++) {**

**rotate\_180(square[row][col], row, col, transposed);**

**}**

**}**

**break;**

**case 'D':**

**for (int row = 0; row < edge; row++) {**

**for (int col = 0; col < edge; col++) {**

**flip\_vertical(square[row][col], row, col, transposed);**

**}**

**}**

**break;**

**case 'E':**

**for (int row = 0; row < edge; row++) {**

**for (int col = 0; col < edge; col++) {**

**flip\_horizontal(square[row][col], row, col, transposed);**

**}**

**}**

**break;**

**case 'Q':**

**launch = false;**

**SetConsoleTextAttribute(hConsole, (WORD)((WHITE << 4) | RED));**

**puts("See you!");**

**SetConsoleTextAttribute(hConsole, (WORD)((WHITE << 4) | BLACK));**

**break;**

**default:**

**SetConsoleTextAttribute(hConsole, (WORD)((WHITE << 4) | RED));**

**puts("Incorrect mode!");**

**SetConsoleTextAttribute(hConsole, (WORD)((WHITE << 4) | BLACK));**

**launch = false;**

**break;**

**}**

**if (launch == true) {**

**puts("Transposed square:");**

**SetConsoleTextAttribute(hConsole, (WORD)((WHITE << 4) | GREEN));**

**for (int row = 0; row < edge; row++) {**

**for (int col = 0; col < edge; col++) {**

**printf("%2d ", transposed[row][col]);**

**}**

**printf("\n");**

**}**

**SetConsoleTextAttribute(hConsole, (WORD)((WHITE << 4) | BLACK));**

**}**

**printf("\n");**

**}**

**return 0;**

**}**

**void rotate\_left(int value, int row, int col, int res\_square[edge][edge]) {**

**int step = edge - 1;**

**res\_square[step - col][row] = value;**

**}**

**void rotate\_right(int value, int row, int col, int res\_square[edge][edge]) {**

**int step = edge - 1;**

**res\_square[col][step - row] = value;**

**}**

**void rotate\_180(int value, int row, int col, int res\_square[edge][edge]) {**

**int step = edge - 1;**

**res\_square[step - row][step - col] = value;**

**}**

**void flip\_vertical(int value, int row, int col, int res\_square[edge][edge]) {**

**int step = edge - 1;**

**res\_square[step - row][col] = value;**

**}**

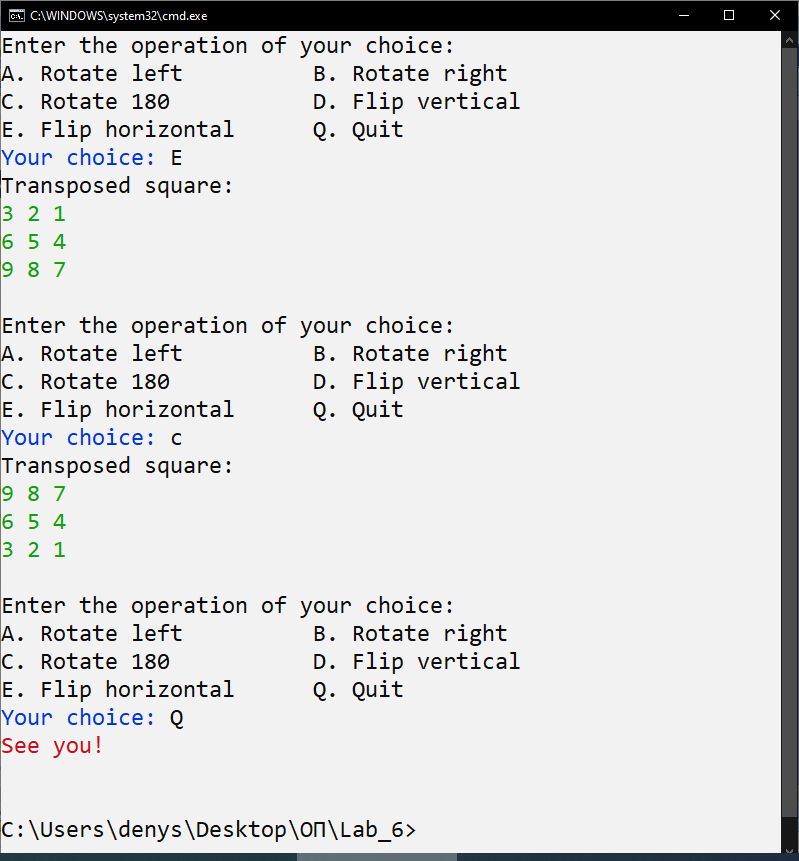
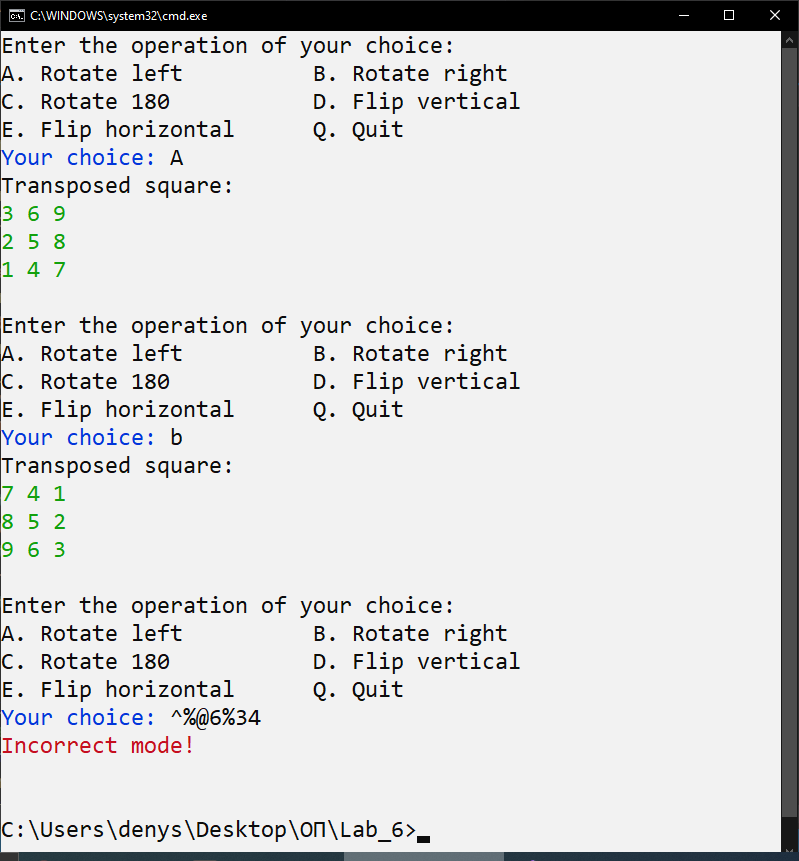
**void flip\_horizontal(int value, int row, int col, int res\_square[edge][edge]) {**

**int step = edge - 1;**

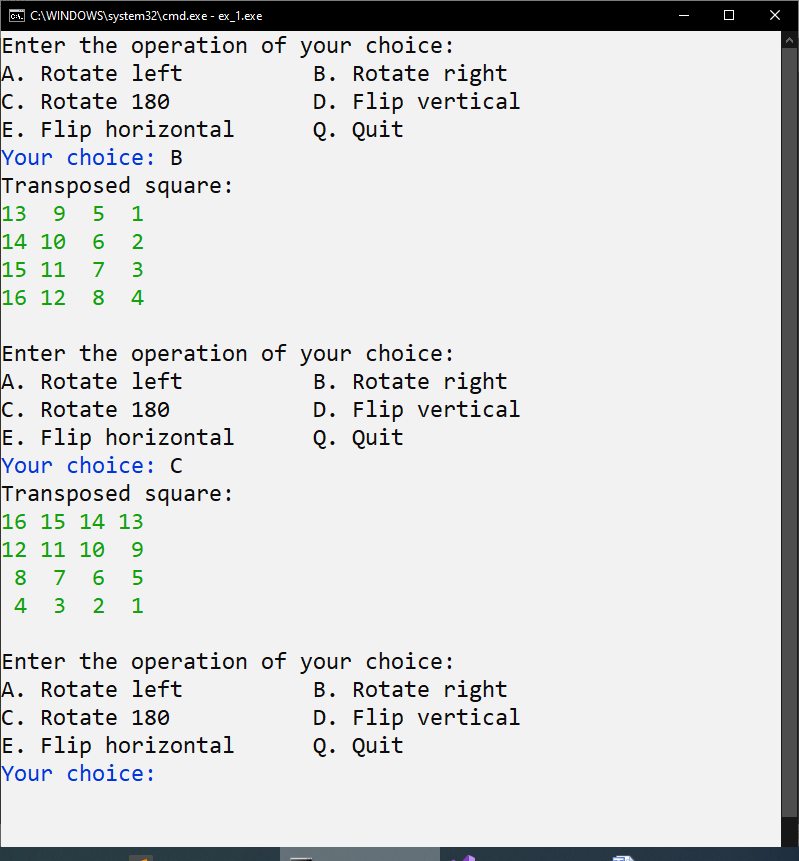
**res\_square[row][step - col] = value;**

**}**

**Результати роботи програми (скриншоти):**



**Завдання № 2:** Напишіть програму, яка буде порівнювати значення двох чисел, при цьому декілька разів запитуючи користувача про введення чисел, які необхідно порівняти. Для порівняння використовуйте умовний оператор «?».



**Код програми:**

**#include <stdio.h>**

**int main(void) {**

**int a, b;**

**while (1) {**

**printf("Enter a: ");**

**scanf("%d", &a);**

**printf("Enter b: ");**

**scanf("%d", &b);**

**fflush(stdin);**

**(a > b) ? puts("a > b") : (a < b ? puts("a < b") : puts("a = b"));**

**printf("\n");**

**printf("To repeat? Enter y/n: ");**

**if (getchar() != 'y') {**

**break;**

**}**

**}**

**return 0;**

**}**

**Результати роботи програми (скриншот):**



**Завдання № 3:** Напишіть програму, яка буде рахувати перші 10 значень функції косинуса. Виконайте програму двома способами: у першому використайте вже реалізовану функції cos() з бібліотеки <cmath.h>, а у другому випадку напишіть свою власну функцію, яка буде рахувати значення косинуса, для цього скористайтесь формулою.

**Код програми:**

**#include <stdio.h>**

**#include <math.h>**

**int factorial(int n);**

**double mycos(double x);**

**int main(void) {**

**double x;**

**printf(" x \t\tmycos(x)\t\tcos(x)\n");**

**for (x = 0; x < 1; x += 0.1) {**

**printf("%.1f\t%.15f\t%.15f\n", x, mycos(x), cos(x));**

**}**

**}**

**int factorial(int n) {**

**return (n < 2) ? 1 : n \* factorial(n - 1);**

**}**

**double mycos(double x) {**

**double sum = 0;**

**for (int i = 0; i < 10; i++)**

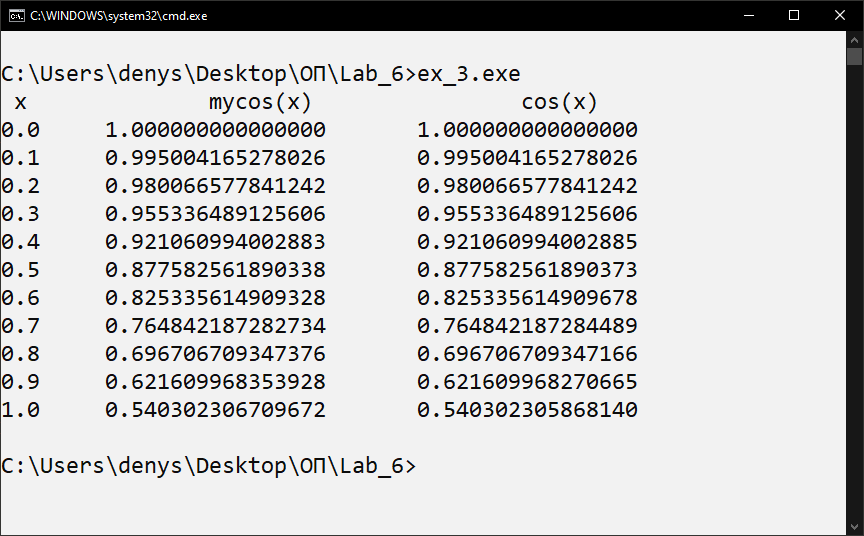
**sum += pow(-1, i) \* pow(x, 2 \* i) / factorial(2 \* i);**

**return sum;**

**}**

**Результати роботи програми:**

Похибка відбувається через заокруглення чисел комп’ютером на проміжному етапі додавання. Саме через це, на деякому повторенні вже ця похибка стає помітною. Числа дуже маленькі, молодші розряди числа у двійковому коді опускаються (при виході за розрядну сітку). З повторенням похибка накопичується.



**Завдання № 4:** Напишіть програму для вирішення задачі Ханойських веж. Задача полягає в тому, що маючи 3 вежі та три кільця різного розміру на першій з них (у вигляді пірамідки), потрібно перемістити ці кільця на останню вежу так, щоб опинились на ній у тому самому порядку.

**Код програми:**

**// The Tower of Hanoi is a mathematical game or puzzle.**

**#include <stdio.h>**

**#define MAX\_NRINGS 64 // max number of rings**

**int st[4][MAX\_NRINGS]; // 1, 2, 3 - the pegs**

**int nr[4]; // number of rings on pegs**

**int nmoves; // number of rings movement**

**// Print current position on pegs**

**void print\_st(void) {**

**for (int i = 1; i <= 3; i++) {**

**printf("\n| ");**

**for (int j = 0; j < nr[i]; j++) {**

**printf("%d", st[i][j]);**

**}**

**}**

**printf("\n");**

**}**

**// Reset**

**void init(int nrings) {**

**for (nr[1] = 0; nrings > 0; nr[1]++, nrings--) {**

**st[1][nr[1]] = nrings;**

**}**

**nr[2] = nr[3] = 0; // All rings on peg#1**

**nmoves = 0;**

**print\_st();**

**}**

**// Move one ring from peg#1 to peg#2**

**void move1(int n1, int n2) {**

**st[n2][nr[n2]++] = st[n1][nr[n1]--];**

**print\_st();**

**nmoves++;**

**}**

**// Move upper nrings from peg i1 to peg i3 i2 as temp**

**// using peg. 1 <= i1, i2, i3 <= 3**

**void hanoi(int nrings, int i1, int i2, int i3) {**

**if (nrings == 1) {**

**move1(i1, i3);**

**}**

**else {**

**hanoi(nrings - 1, i1, i3, i2);**

**move1(i1, i3);**

**hanoi(nrings - 1, i2, i1, i3);**

**}**

**}**

**int main(void) {**

**int nrings;**

**printf("Rings number: ");**

**scanf("%d", &nrings);**

**hanoi(nrings, 1, 2, 3);**

**printf("All rings were moved.\n");**

**printf("Moves: %d\n", nmoves);**

**return 0;**

**}**

**Результати роботи програми (скриншот):**

