**SOURCE CODE**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <time.h>**

**#define MAX\_N 700**

**int main()**

**{**

**int i, j, n;**

**printf("Enter The Number Of Relation Matrices (n): ");**

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

**if (n > MAX\_N)**

**{**

**printf("Matrix Size Exceeds The Maximum Allowed Size.\n");**

**return 1;**

**}**

**FILE \*outputFile;**

**outputFile = fopen("relations.txt", "w"); // Open a file for writing**

**if (outputFile == NULL)**

**{**

**printf("Error Opening The Output File.\n");**

**return 1;**

**}**

**srand(time(0)); // Seed the random number generator with the current time**

**// Declare the matrix using the correct size 'n'**

**int mat[MAX\_N][MAX\_N];**

**for (int a = 0; a < n; a++)**

**{**

**fprintf(outputFile, "Matrix :\n");**

**for (i = 0; i < n; i++)**

**{**

**for (j = 0; j < n; j++)**

**{**

**if (i == j)**

**{**

**mat[i][j] = 0; // Diagonal elements are set to 0 (no self-relations)**

**}**

**else**

**{**

**mat[i][j] = rand() % 2; // Generate random 0 or 1**

**mat[j][i] = mat[i][j]; // Ensure symmetry**

**}**

**fprintf(outputFile, "%d ", mat[i][j]); // Note the space after %d**

**}**

**fprintf(outputFile, "\n");**

**}**

**struct timespec start\_time, end\_time;**

**clock\_gettime(CLOCK\_MONOTONIC, &start\_time);**

**int symmetric = 1;**

**int antisymmetric = 1;**

**int transitive = 1;**

**int equivalence = 0; // Initialize equivalence to 0**

**int function = 1;**

**// Check for symmetry and anti-symmetry**

**for (i = 0; i < n; i++)**

**{**

**for (j = 0; j < n; j++)**

**{**

**if (mat[i][j] != mat[j][i])**

**{**

**symmetric = 0;**

**}**

**if (mat[i][j] == mat[j][i] && i != j)**

**{**

**antisymmetric = 0;**

**}**

**}**

**}**

**// Check for transitivity**

**for (i = 0; i < n; i++)**

**{**

**for (j = 0; j < n; j++)**

**{**

**if (mat[i][j] == 1)**

**{**

**for (int k = 0; k < n; k++)**

**{**

**if (mat[j][k] == 1 && mat[i][k] != 1)**

**{**

**transitive = 0;**

**}**

**}**

**}**

**}**

**}**

**// Check for equivalence**

**if (symmetric && transitive)**

**{**

**equivalence = 1;**

**}**

**// Check for function**

**for (j = 0; j < n; j++)**

**{**

**int count = 0;**

**for (i = 0; i < n; i++)**

**{**

**if (mat[i][j] == 1)**

**{**

**count++;**

**if (count > 1)**

**{**

**function = 0;**

**break;**

**}**

**}**

**}**

**}**

**clock\_gettime(CLOCK\_MONOTONIC, &end\_time);**

**double time\_taken = (end\_time.tv\_sec - start\_time.tv\_sec) \* 1000.0;**

**time\_taken += (end\_time.tv\_nsec - start\_time.tv\_nsec) / 1000000.0;**

**fprintf(outputFile, "Symmetric: %s\n", symmetric ? "yes" : "no");**

**fprintf(outputFile, "Anti-symmetric: %s\n", antisymmetric ? "yes" : "no");**

**fprintf(outputFile, "Transitive: %s\n", transitive ? "yes" : "no");**

**fprintf(outputFile, "Equivalence: %s\n", equivalence ? "yes" : "no");**

**fprintf(outputFile, "Function: %s\n", function ? "yes" : "no");**

**fprintf(outputFile, "Execution time: %lf ms\n", time\_taken);**

**}**

**fclose(outputFile); // Close the output file**

**return 0;**

**}**