```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define MAXN 5000

int graph[MAXN][MAXN];

int main() {
    int sizes[] = {1000, 2000, 3000, 4000, 5000};
    int num_sizes = 5;

    srand(time(NULL));

    for (int s = 0; s < num_sizes; s++) {
        int n = sizes[s];

        printf("\033[0;32mEnter the number of vertices: %d\n\033[0m", n);

    for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                graph[i][j] = rand() % 2;
            }
        }

        clock_t start, end;
        start = clock();

        long long indeg_sum = 0, outdeg_sum = 0;

        for (int i = 0; i < n; i++) {
            int indeg = 0, outdeg = 0;

            for (int j = 0; j < n; j++) {
                outdeg += graph[i][j];
                indeg += graph[j][i];
            }

            indeg_sum += indeg;
            outdeg_sum += outdeg;
        }

        end = clock();

        double cpu_time_used = ((double)(end - start)) / CLOCKS_PER_SEC * 1000;
// ms
```

```
    printf("\033[0;36m");

    printf("Sum of in-degrees : %lld\n", indeg_sum);
    printf("Sum of out-degrees : %lld\n", outdeg_sum);

if (indeg_sum == outdeg_sum)
    printf("Sum of in-degrees & out-degrees are equal.\n");
else
    printf("Sum of in-degrees & out-degrees are NOT equal.\n");

    printf("The computational time : %.3f ms\n\n", cpu_time_used);

    printf("\033[0m");
    }

    return 0;
}
```