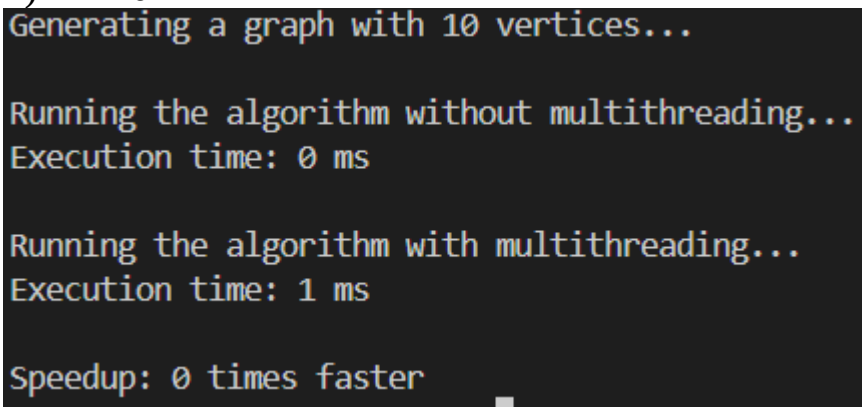
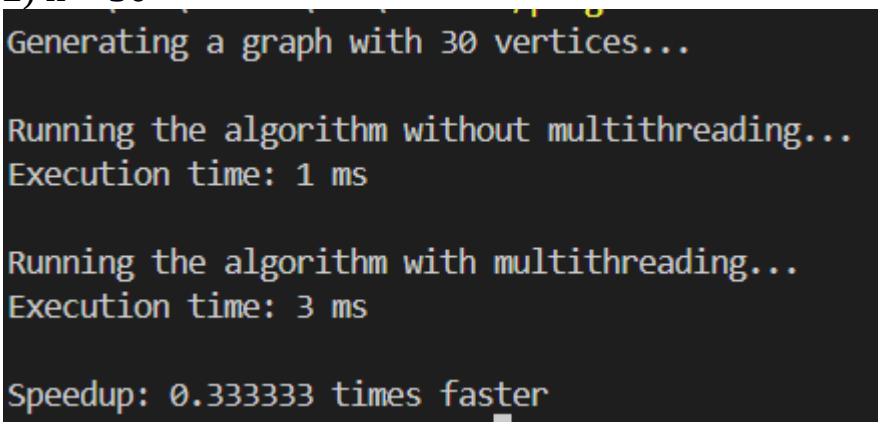
Лабораторна робота №3а
Патерни програмування
Звіт
Виконав студент групи ІПС-21
Міцкевич Костянтин

Посилання на репозиторій: https://github.com/Kirolen/OOP-Lab3
У рамках цієї лабораторної було реалізовано алгоритм Джонсона з використанням мультипоточності та без неї на мові c++.
Для зручності будемо проводити бенчмарк з різною кількістю вершин зі щільністю ребер 0.55, та на кожному кроці буде збільшувати кількість вершин в 3 рази.
1) n = 10



2) n = 30



3) n = 90

```
Generating a graph with 90 vertices...

Running the algorithm without multithreading...
Execution time: 18 ms

Running the algorithm with multithreading...
Execution time: 18 ms

Speedup: 1 times faster
```

4) n = 270

```
Generating a graph with 270 vertices...

Running the algorithm without multithreading...
Execution time: 342 ms

Running the algorithm with multithreading...
Execution time: 230 ms

Speedup: 1.48696 times faster
```

5) n = 810

```
Generating a graph with 810 vertices...

Running the algorithm without multithreading...
Execution time: 7848 ms

Running the algorithm with multithreading...
Execution time: 5659 ms

Speedup: 1.38682 times faster
```

6) n = 1000

```
Generating a graph with 1000 vertices...

Running the algorithm without multithreading...
Execution time: 29494 ms

Running the algorithm with multithreading...
Execution time: 14192 ms

Speedup: 2.07821 times faster
```

Отже, бачимо що на малих графах мультипоточність працює повільніше ніж звичайна функція. Лише на більших графах мультипоточність працює швидше. Пік був у графах з 1000 вершин. Тоді мультипоточність працює в 2 рази швидше.