This project aims to observe the difference in performance across variations of BFS, including sequential, future, and thread version, applied on problems.

Note: For some unknown reasons, running all tests continuously is quite unstable (Some tests will take longer or faster than expected).
For accuracy, thus, the result of this project will be based on measuring each test one by one.

Test specifications:
- Each test is repeating the process 10 times and get the average result
- Test 1.1 - 1.3: Running BFS to solve a maze problem.
- Test 2.1 - 2.3: Running BFS to find related synonyms in the database.
- Test x.1, x.2, x.3 means sequential, Future, Thread version respectively.

First Observation: By running tests # 1.1, 1.2, 1.3, we can observe from the result that test 1.1 performs better than another two concurrent versions. And the future and thread version both have about the same performance.

```
Test 1.1 seq: result=Some(urrrrruurrrrrdddrrurrrrrrrrrrruurrrrrdddrruuurrrrrrrrrddrruurrrrrdddrrrrrrrrurrrrruurrrrrdddrruuurrr), time=0.034609s
Test 1.2 fut: result=Some(urrrrruurrrrrdddrrurrrrrrrrrrruurrrrrdddrruuurrrrrrrrrddrruurrrrrdddrrrrrrrrurrrrruurrrrrddrdrruuurrr), time=0.071792s
Test 1.3 thr: result=Some(rurrrrururrrrrdddrrrrrurrrrrrrrrruurrrrdrddrruuurrrrrrrrrddrruurrrrrdddrrrrrrrrurrrrruurrrrrdddrruuurrr), time=0.073422s
```

Second Observation: By running test # 2.1, 2.2, 2.3, we can observe from the result that test 2.2 and 2.3 performs better than the sequential version. And the future and thread version both have about the same performance.

```
Test 2.1 seq: result=Some(List(illogical, unreasonable, unrealistic, unreal, synthetic, logical)), time=1.931068s
Test 2.2 fut: result=Some(List(illogical, unreasonable, unrealistic, unreal, synthetic, logical)), time=1.435538s
Test 2.3 thr: result=Some(List(illogical, unreasonable, unrealistic, unreal, synthetic, logical)), time=1.413342s
```

Conclusion: According to both observations, shape of the graph really matters the performance across variations of BFS. I can conclude the result to a few important points including
- The sequential version runs faster than the concurrent version when the graph is not wide (i.e each vertex has only few neighbors), as we can see that solving a maze problem, each vertex has only up to 6 neighbors.
- The concurrent version runs faster than the sequential version when the graph is wide (i.e each vertex has many neighbors), as it is worth picking up a new thread to compute for all adjacent neighbors.
- Both concurrent versions overall have about the same performance (Based on my machine).