

# Systems Programming

## Coursework 2a

1.

My code works as is expected and is implemented as a multithreaded solution. As expected there is no output when running:

“../dependencyDiscoverer \*.y \*.l \*.c | diff – output”

2.

```
-bash-4.2$ pwd
/users/level3/
-bash-4.2$ make sequentialDependencyDiscoverer
clang++ -Wall -Werror -std=c++17 -o sequentialDependencyDiscoverer sequentialDependencyDiscoverer.cpp -lpthread
-bash-4.2$ make dependencyDiscoverer
clang++ -Wall -Werror -std=c++17 -o dependencyDiscoverer dependencyDiscoverer.cpp -lpthread
-bash-4.2$ time ./sequentialDependencyDiscoverer -Itest test/*.c test/*.l test/*.y > temp

real    0m0.074s
user    0m0.007s
sys     0m0.025s
-bash-4.2$ export CRAWLER_THREADS=1
-bash-4.2$ time ./dependencyDiscoverer -Itest test/*.c test/*.l test/*.y > temp

real    0m0.050s
user    0m0.012s
sys     0m0.015s
```

3.

```
-bash-4.2$ pwd
/users/level3/
-bash-4.2$ export CRAWLER_THREADS=1
-bash-4.2$ time ./dependencyDiscoverer -Itest test/*.c test/*.l test/*.y > temp

real    0m0.082s
user    0m0.013s
sys     0m0.018s
-bash-4.2$ export CRAWLER_THREADS=2
-bash-4.2$ time ./dependencyDiscoverer -Itest test/*.c test/*.l test/*.y > temp

real    0m0.028s
user    0m0.009s
sys     0m0.018s
-bash-4.2$ export CRAWLER_THREADS=3
-bash-4.2$ time ./dependencyDiscoverer -Itest test/*.c test/*.l test/*.y > temp

real    0m0.025s
user    0m0.013s
sys     0m0.018s
-bash-4.2$ export CRAWLER_THREADS=4
-bash-4.2$ time ./dependencyDiscoverer -Itest test/*.c test/*.l test/*.y > temp

real    0m0.021s
user    0m0.009s
sys     0m0.020s
-bash-4.2$ export CRAWLER_THREADS=6
-bash-4.2$ time ./dependencyDiscoverer -Itest test/*.c test/*.l test/*.y > temp

real    0m0.019s
user    0m0.010s
sys     0m0.021s
-bash-4.2$ export CRAWLER_THREADS=8
-bash-4.2$ time ./dependencyDiscoverer -Itest test/*.c test/*.l test/*.y > temp

real    0m0.018s
user    0m0.011s
sys     0m0.024s
```

CRAWLER_ THREADS	1	2	3	4	6	8
	Elapsed Time	Elapsed Time	Elapsed Time	Elapsed Time	Elapsed Time	Elapsed Time
Execution 1	0.053s	0.030s	0.024s	0.021s	0.019s	0.019s
Execution 2	0.050s	0.030s	0.025s	0.022s	0.020s	0.019s
Execution 3	0.053s	0.031s	0.024s	0.021s	0.020s	0.019s
Median	0.053s	0.030s	0.024s	0.021s	0.020s	0.019s

From this experiment we can conclude that the performance gains fall drastically when having more than 4 threads this is due to the amount of time it take to create the threads which negates the performance gain from running it with threads.