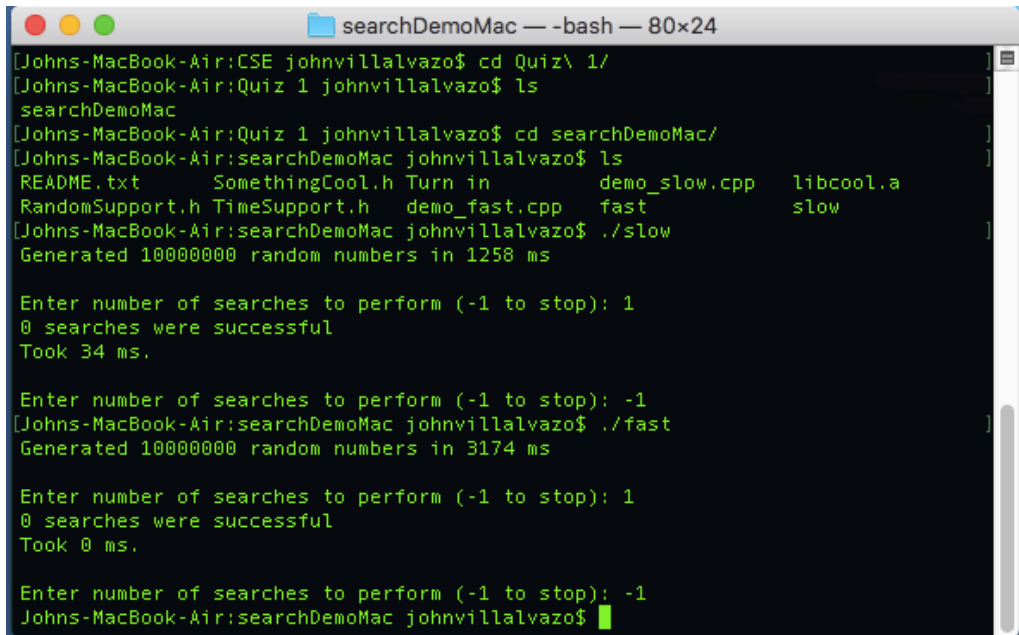


Quiz 1

- 1) The demo_fast.cpp file was the faster of the two files. Demo_fast.cpp was able to compute input 1 in 0 ms whereas the demo_slow.cpp file took 32 ms on my machine.



```
[Johns-MacBook-Air:CSE johnvillalvazo$ cd Quiz\ 1/
[Johns-MacBook-Air:Quiz 1 johnvillalvazo$ ls
searchDemoMac
[Johns-MacBook-Air:Quiz 1 johnvillalvazo$ cd searchDemoMac/
[Johns-MacBook-Air:searchDemoMac johnvillalvazo$ ls
README.txt      SomethingCool.h  Turn in         demo_slow.cpp   libcool.a
RandomSupport.h TimeSupport.h    demo_fast.cpp   fast            slow
[Johns-MacBook-Air:searchDemoMac johnvillalvazo$ ./slow
Generated 10000000 random numbers in 1258 ms

Enter number of searches to perform (-1 to stop): 1
0 searches were successful
Took 34 ms.

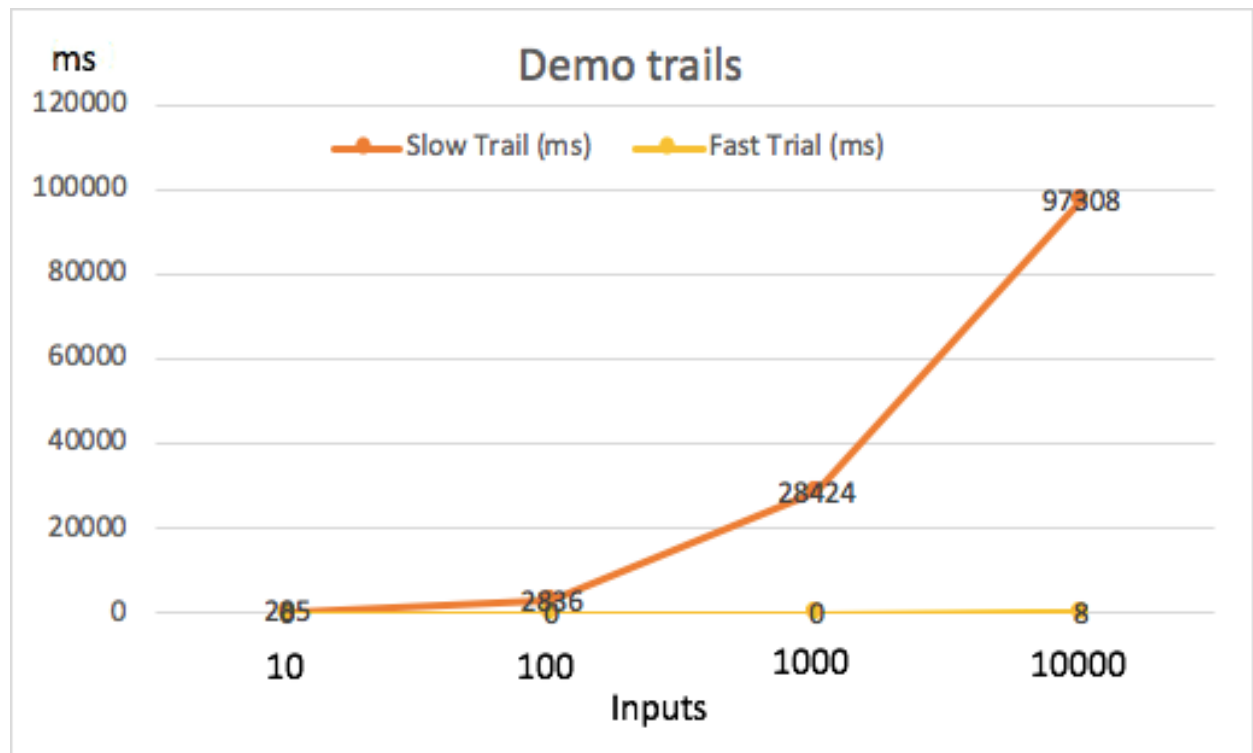
Enter number of searches to perform (-1 to stop): -1
[Johns-MacBook-Air:searchDemoMac johnvillalvazo$ ./fast
Generated 10000000 random numbers in 3174 ms

Enter number of searches to perform (-1 to stop): 1
0 searches were successful
Took 0 ms.

Enter number of searches to perform (-1 to stop): -1
Johns-MacBook-Air:searchDemoMac johnvillalvazo$
```

2) demo_fast.cpp took longer to generate all random numbers whereas demo_slow.cpp was slower. Demo_fast.cpp took 3174 ms to generate all numbers whereas demo_slow.cpp took 1258 ms. I think the reason this happened is because in the structs that they programs use, the trade-off for a faster allocation and generation of numbers would probably be a slower search time and vice-versa.

3) There is a sizeable time difference with the performance of these two programs. The demo_slow.cpp program takes 27646 ms to compute the input of 1000 whereas demo_fast.cpp takes 0 ms on my machine to compute to the same number.



4) For demo_slow, the maximum number that can be inputted and computed before reaching 1 second is the number 3 whereas for demo_fast on my machine was 4300 before the time it took to compute was 1 second.