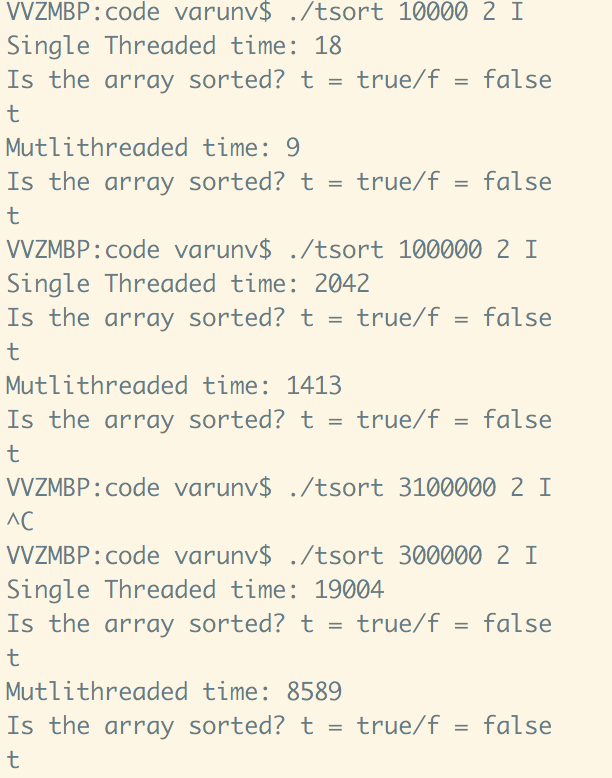
**Varun Ved**

**CSC 139-04**

**Multithreaded Sort Writeup**

**Timings + Screenshots**

**1. Run InsertionSort using two threads with array sizes 10K, 100K and 300K.**

VVZMBP:code varunv$ ./tsort **10000** 2 I

**Single Threaded time: 18**

Is the array sorted? t = true/f = false

t

**Mutlithreaded time: 9**

Is the array sorted? t = true/f = false

t

VVZMBP:code varunv$ ./tsort **100000** 2 I

**Single Threaded time: 2042**

Is the array sorted? t = true/f = false

t

**Mutlithreaded time: 1413**

Is the array sorted? t = true/f = false

t

VVZMBP:code varunv$ ./tsort **300000** 2 I

**Single Threaded time: 19004**

Is the array sorted? t = true/f = false

t

**Mutlithreaded time: 8589**

Is the array sorted? t = true/f = false

t

**2. Run InsertionSort using four threads with an array size of 100K.**

**VVZMBP:code varunv$ ./tsort 100000 4 I**

**Single Threaded time: 1034**

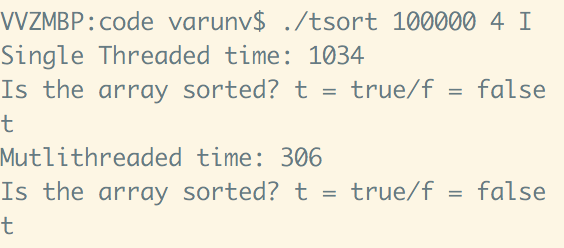
Is the array sorted? t = true/f = false

t

**Mutlithreaded time: 306**

Is the array sorted? t = true/f = false

t



**3. Run QuickSort using two threads with array sizes 1M, 10M and 100M.**

VVZMBP:code varunv$ ./tsort **1000000** 2 Q

**Single Threaded time: 132**

Is the array sorted? t = true/f = false

t

**Mutlithreaded time: 48**

Is the array sorted? t = true/f = false

t

VVZMBP:code varunv$ ./tsort **10000000** 2 Q

**Single Threaded time: 1675**

Is the array sorted? t = true/f = false

t

**Mutlithreaded time: 550**

Is the array sorted? t = true/f = false

t

VVZMBP:code varunv$ ./tsort **100000000** 2 Q

**Single Threaded time: 17322**

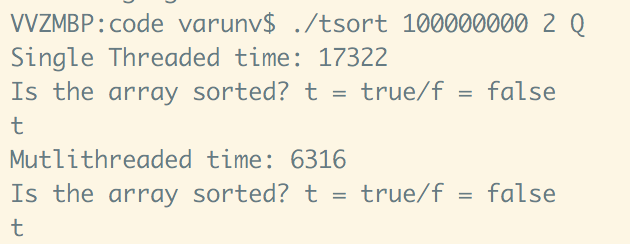
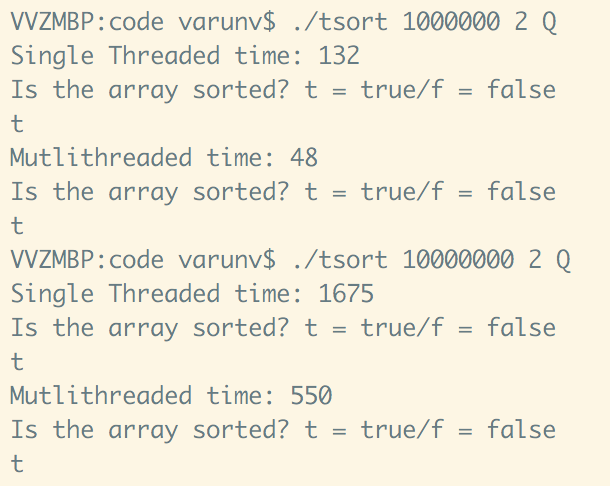
Is the array sorted? t = true/f = false

t

**Mutlithreaded time: 6316**

Is the array sorted? t = true/f = false

t



**4. Run QuickSort using four threads with an array size of 10M**

VVZMBP:code varunv$ ./tsort **10000000** 4 Q

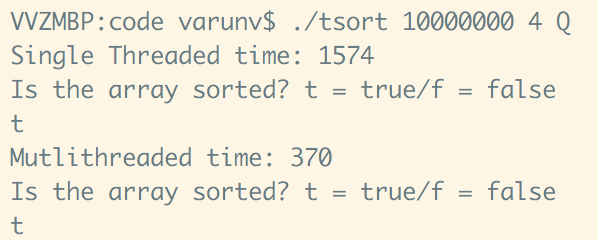
**Single Threaded time: 1574**

Is the array sorted? t = true/f = false

t

**Mutlithreaded time: 370**

Is the array sorted? t = true/f = false

t

**Tables**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| array size | 2 Threads Insertion | 4 threads Insertion | 2 Threads QS | 4 Threads QS |
| 10k | 18/9 |  |  |  |
| 100k | 2042/1413 | 1034/306 |  |  |
| 300k | 19004/8589 |  |  |  |
| 1m |  |  | 132/48 |  |
| 10m |  |  | 1675/550 | 1574/370 |
| 100m |  |  | 17322/6316 |  |

**2 thread insertion**

|  |  |
| --- | --- |
| array size | 2 Threads Insertion |
| 10k | 18/9 |
| 100k | 2042/1413 |
| 300k | 19004/8589 |
| 1m |  |
| 10m |  |
| 100m |  |

**4 thread insertion**

|  |  |
| --- | --- |
| array size | 4 threads Insertion |
| 10k |  |
| 100k | 1034/306 |
| 300k |  |
| 1m |  |
| 10m |  |
| 100m |  |

**2 threaded qs**

|  |  |
| --- | --- |
| array size | 2 Threads QS |
| 10k |  |
| 100k |  |
| 300k |  |
| 1m | 132/48 |
| 10m | 1675/550 |
| 100m | 17322/6316 |

**4 threaded qs**

|  |  |
| --- | --- |
| array size | 4 Threads QS |
| 10k |  |
| 100k |  |
| 300k |  |
| 1m |  |
| 10m | 1574/370 |
| 100m |  |

**Write-up**

As we can see from the table, different algorithms and array sizes reacted differently in relation to their algorithms provided. For insertion sort, doubling the number of threads made it run a bit more than twice as fast for the single threaded time on 100k elements, however on the multithread, it ran 4x as fast! Insertion sort didn't help with threads nearly as much as it did with quicksort. Quicksort however reacted very well to multiple threads as it performed faster than insertion on 10x as much data. However, adding more threads didn't help too much.