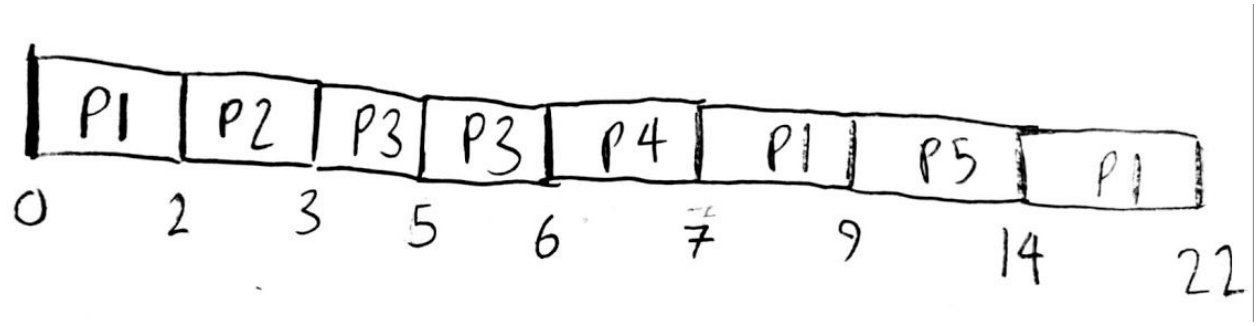
Q1

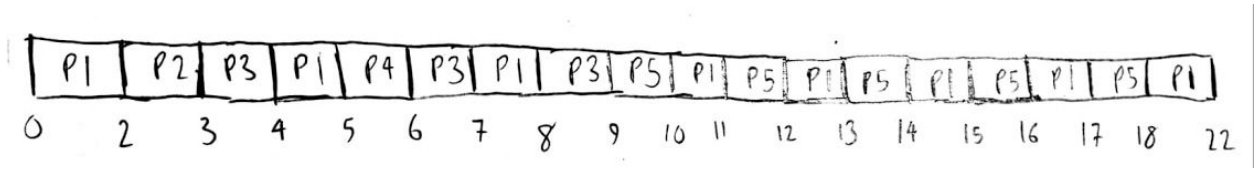
The CPU utilization is 89.99%.

Q2



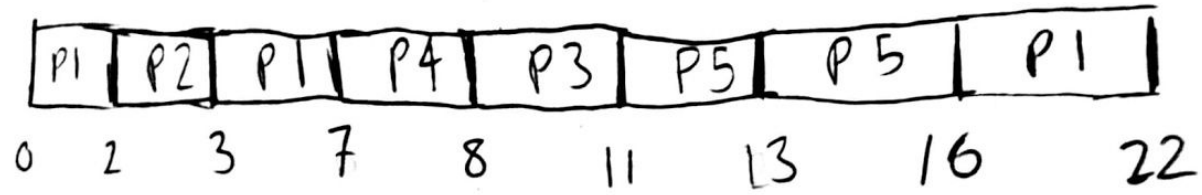

Average waiting time: 2.2 ms

Q3




There are 17 context switches when round-robin is used with a 1 ms quantum.

Q4



Q6

Timings for `medium.txt`

| Number of Threads | Observed timing | Observed speedup compared to original | Expected speedup |
|---|---|---|---|
| Original program | 17.450 s | 1.0 | 1.0 |
| 1 | 17.961 s | 0.97 | 1.0 |
| 2 | 10.013 s | 1.74 | 2.0 |
| 3 | 11.168 s | 1.56 | 3.0 |
| 4 | 9.624 s | 1.81 | 4.0 |
| 8 | 9.321 s | 1.87 | 8.0 |
| 16 | 9.990 s | 1.74 | 16.0 |

Timings for `hard.txt`

| Number of Threads | Observed timing | Observed speedup compared to original | Expected speedup |
|---|---|---|---|
| Original program | 5.936 s | 1.0 | 1.0 |
| 1 | 6.126 s | 0.97 | 1.0 |
| 2 | 6.054 s | 0.98 | 2.0 |
| 3 | 6.019 s | 0.99 | 3.0 |
| 4 | 6.036 s | 0.98 | 4.0 |
| 8 | 6.101 s | 0.97 | 8.0 |
| 16 | 6.070 s | 0.98 | 16.0 |

Timings for `hard2.txt`

| Number of Threads | Observed timing | Observed speedup compared to original | Expected speedup |
|---|---|---|---|
| Original program | 5.961 s | 1.0 | 1.0 |
| 1 | 6.147 s | 0.97 | 1.0 |
| 2 | 6.058 s | 0.98 | 2.0 |
| 3 | 6.109 s | 0.98 | 3.0 |
| 4 | 6.071 s | 0.98 | 4.0 |
| 8 | 6.052 s | 0.98 | 8.0 |
| 16 | 6.022 s | 0.99 | 16.0 |

For `medium.txt`, I think the reason the multithreaded program doesn't continue any faster after two threads is because one or more threads is maybe doing more work than the other threads. So those threads that are behind are holding up the other ones that are ahead. For `hard.txt` and `hard2.txt`, they were all slightly slower than the multithreaded program, which makes me believe that there one thread that is holding the ones that are ahead back. I believe they are all slightly slower also because the multithreaded program is using C++ while the original is using C, and C possibly has a slightly faster run time than C++.