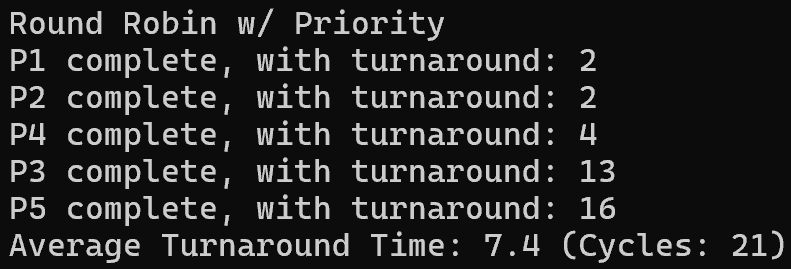
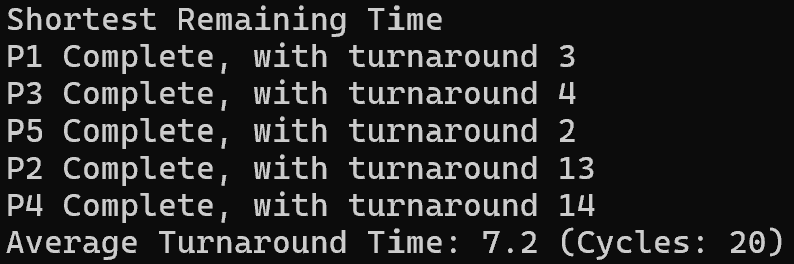


SJF was an easy one to implement, as it required no sorting of any kind. The hardest part of it was finding the process with the shortest burst time, which could have been done with a single for loop. I designed the algorithm in a way where the for loop for finding the shortest burst process couldn’t even be run with an empty list, so I had to fix that by giving the variable “lowest” an absurd and (hopefully) impossible-to-reach initial value of 999999999. From the turnaround time in the output, and from designing the algorithm myself, I can see how with modern computers with thousands of processes could be slowed down using this approach.



This was a very difficult algorithm to write. I initially thought priority wouldn’t have a big of an impact on the turnaround time for RR as it uses a queue system, but after comparing the output with the online calculator ([Process Scheduling Solver](https://process-scheduling-solver.boonsuen.com/)) there were noticeable differences. The hardest challenge for RR was figuring out how to deal with the priority of processes in between the first and last positions of the queue. I assumed the processes in a queue should be lined up from highest priority to the lowest, but then it would not make sense as a RR algorithm. I ended only using the first position to determine where the newly arriving processes are sent. I may have been overthinking this whole thing, but if this algorithm is more complicated than what I did here, how is the CPU processing this algorithm for every incoming process? I don’t think this algorithm would have been viable before the modern, powerful CPUs.



After going through with RR, I decided to finish it up with the algorithm I’m familiar with, which was SRT. With what we learned in class, it was easy to plan out the overall layout of the algorithm. A challenge I faced here was figuring out how to use a variable from a class as a target variable for sorting the list. Thankfully, there was a stack overflow post ([c# - List<T> OrderBy Alphabetical Order - Stack Overflow](https://stackoverflow.com/questions/188141/listt-orderby-alphabetical-order)) on the topic, and I was able to proceed to the rest of the algorithm. I had almost no grasp of which algorithms were more efficient than others (except on paper), but as SRT is being compared with SJF and RR, I can see that it is the fastest algorithm here. I can imagine the differences when there are thousands of processes to process.