**CS202 - Advanced Operating System**

**LAB 2**

* Tejas Milind Deshpande , sid- 862393211
* Satya sri Nandan Paritala , sid - 862395423
* Nunna Lakshmi Saranya , sid- 862394536

**A.Demo Video**

<https://drive.google.com/file/d/1CIJxY-9hX94Yeif4w31NzWQc2VO1xMSR/view?usp=share_link>

**B.List of all modified files-**

* MAKEFILE
* kernel/defs.h
* kernel/proc.c
* kernel/proc.h
* kernel/syscall.c
* kernel/syscall.h
* kernel/sysproc.c
* user/user.h
* user/usys.pl
* New file for testing - lab2.c

**C. Changes and screenshots:-**

**Part 1: System Calls**

We implemented 2 system calls -

* int sched\_statistics(void);This system call prints, for each process, 1) PID, 2) name in a parenthesis, 3) the ticket value, and 4) the number of times it has been scheduled to run
* int sched\_tickets(int); This system call sets the caller process’s ticket value to the given parameter

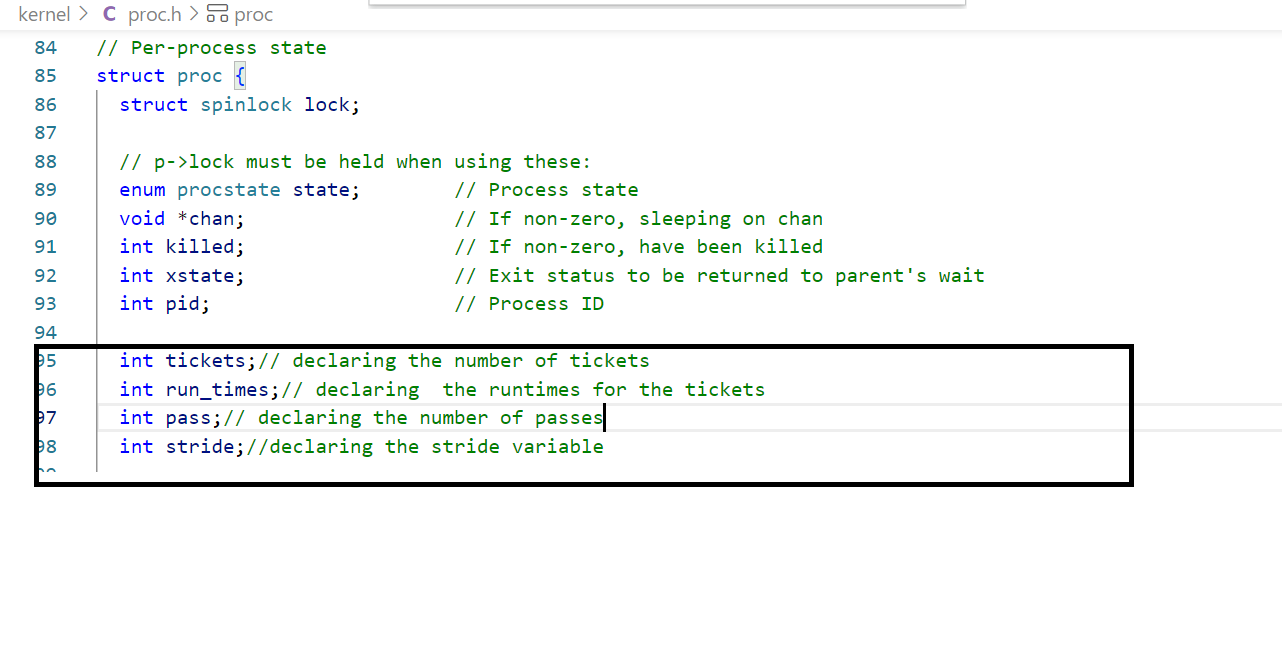
**kernel/defs.h**

Declaring new Kernel functions to assign tickets and print process statistics:



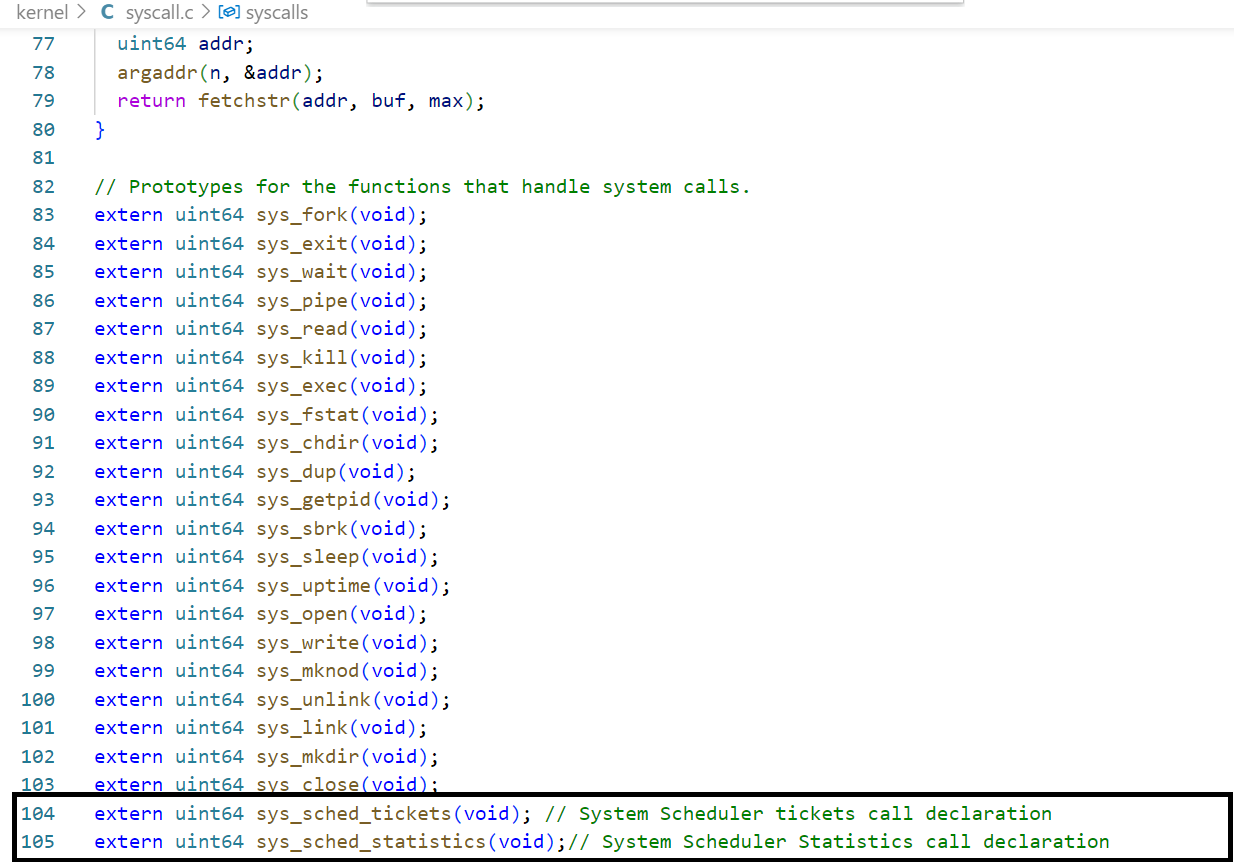
**kernel/proc.h**

Adding tickets, runt\_times(ticks), pass and stride to process structure to keep track of tickets in Lottery scheduling, tickets, stride and pass in Stride scheduling and process runtime in ticks.



**kernel/syscall.c**

In order to initiate the system calls from a user-level program, we expose the sys\_sched\_tickets & sys\_sched\_statistics call as extern.

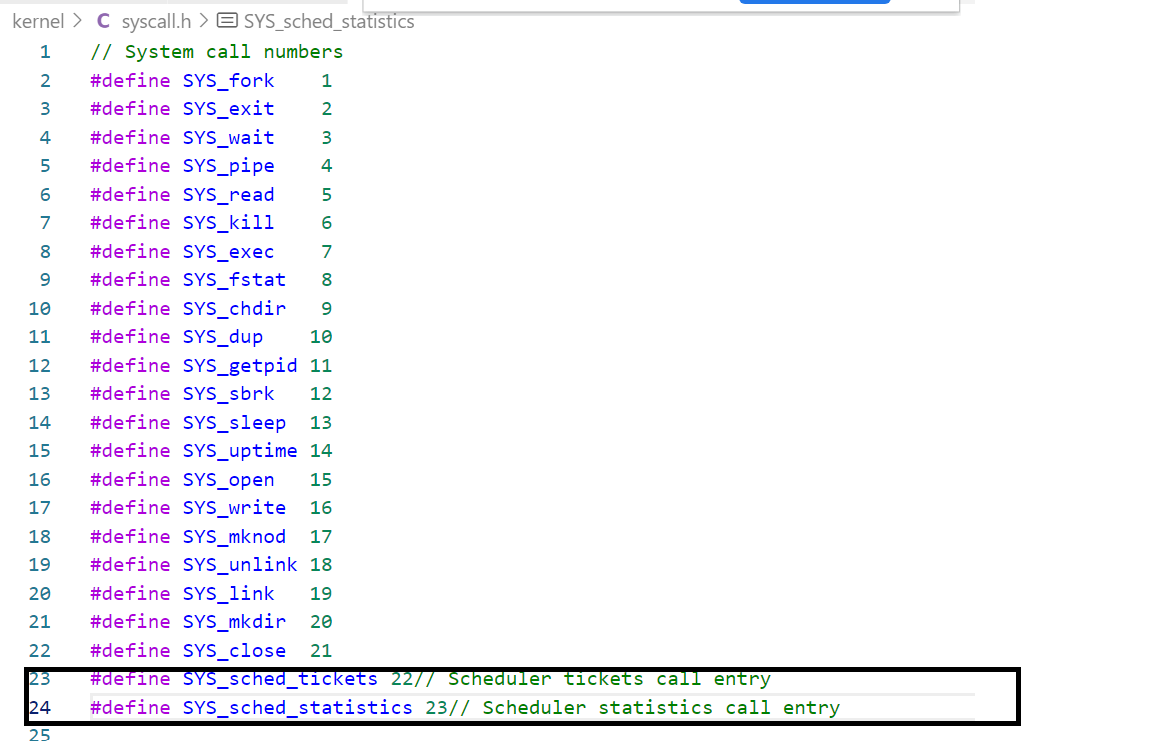


Updated this file with the pointer:



**kernel/syscall.h**

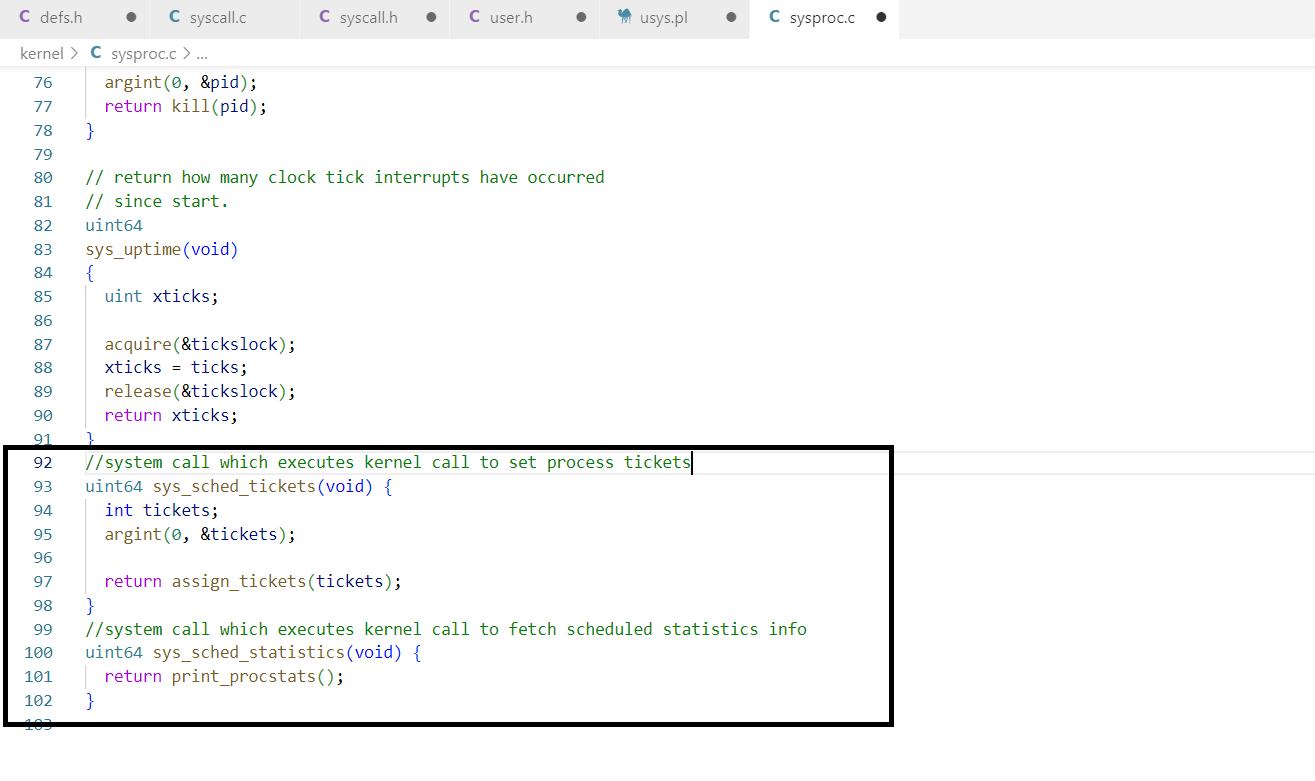
Defining new system call number for scheduler tickets and scheduled statistics:



**kernel/sysproc.c:**

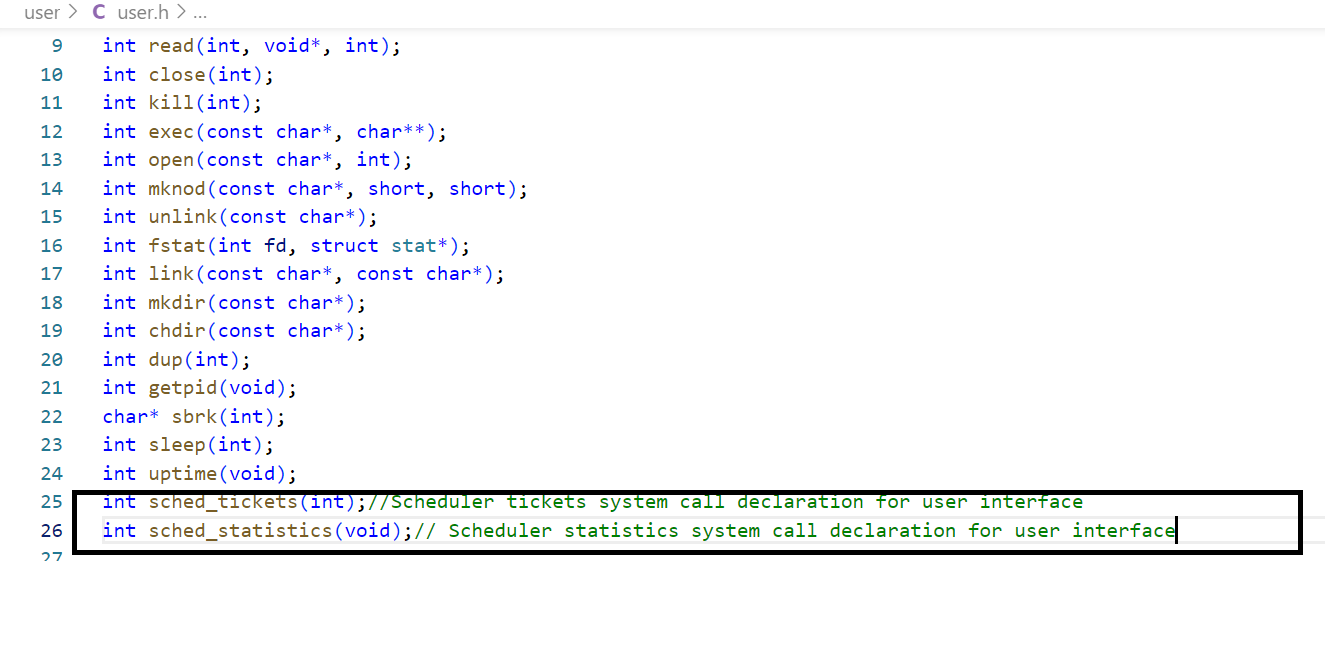
A new sys\_sched\_tickets() function which takes tickets as integer argument using argint() system call from user level and assigns it to process in kernel call.

A new sys\_sched\_statistics() call to print process statistics.

****

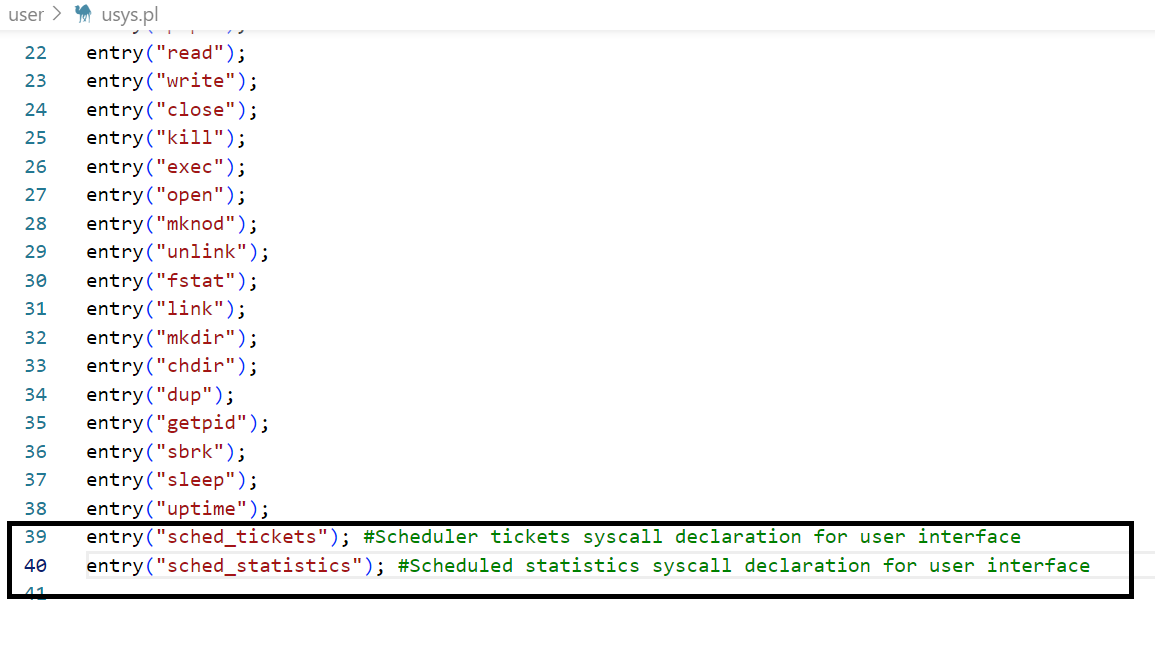
**user/user.h**

Exposing the sched\_statistics() and sched\_tickets() call in user space sys call interface

****

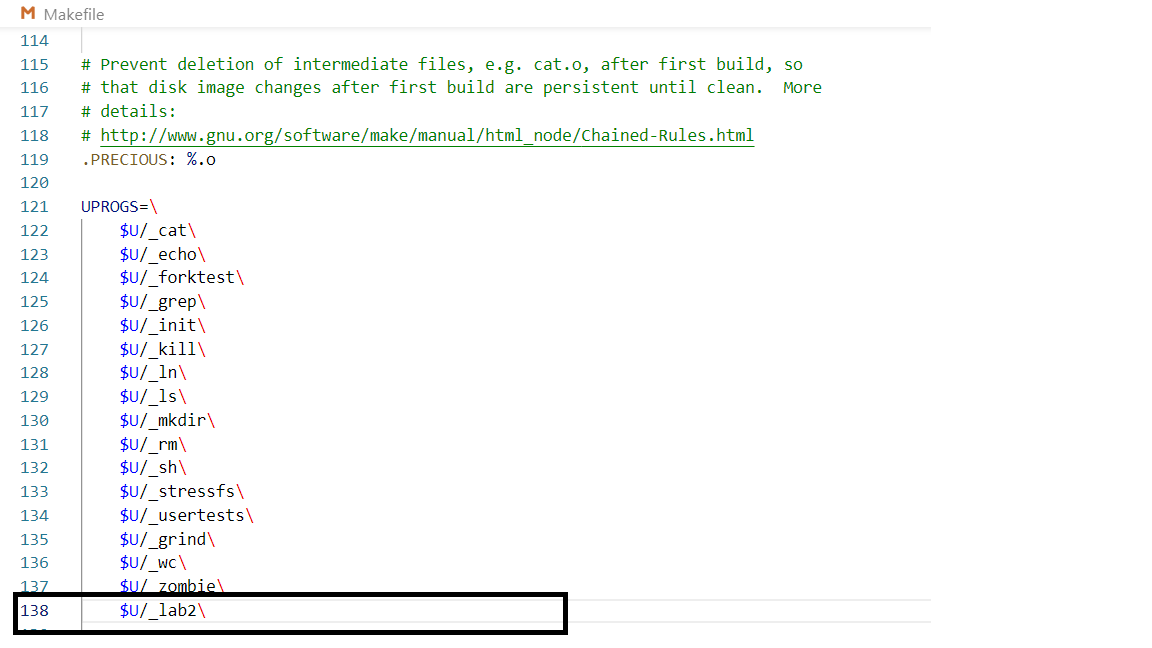
**Usys.pl**

Exposing the sched\_statistics() and sched\_tickets() call in user space sys call interface.

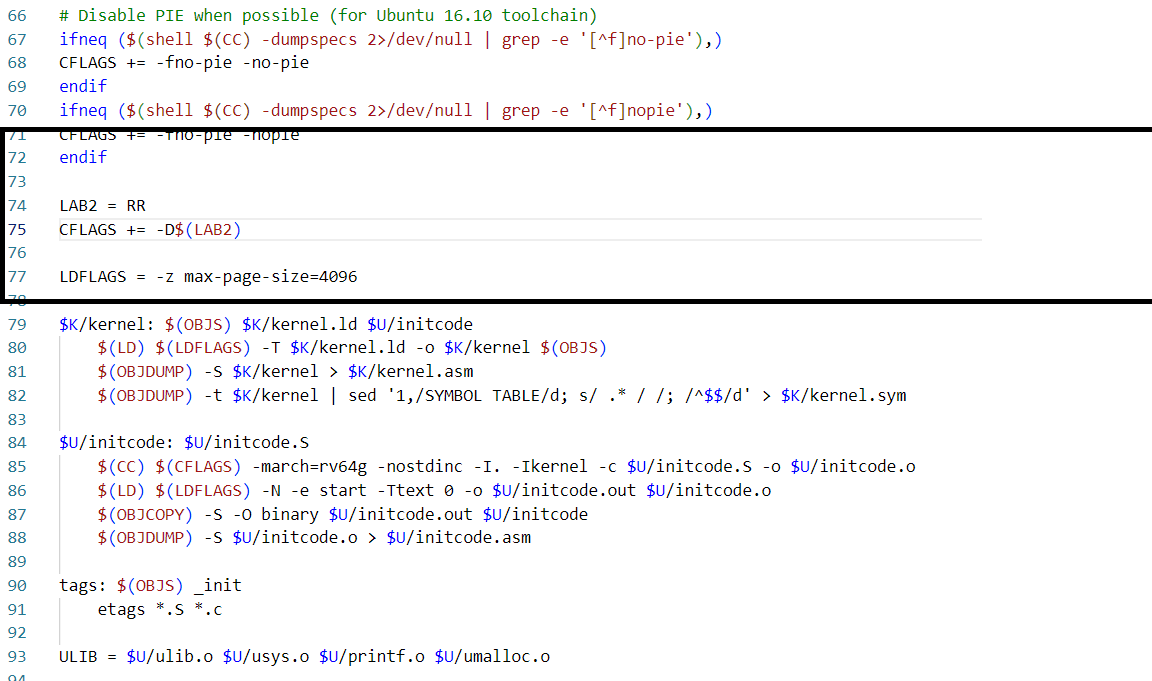


**MakeFile:**

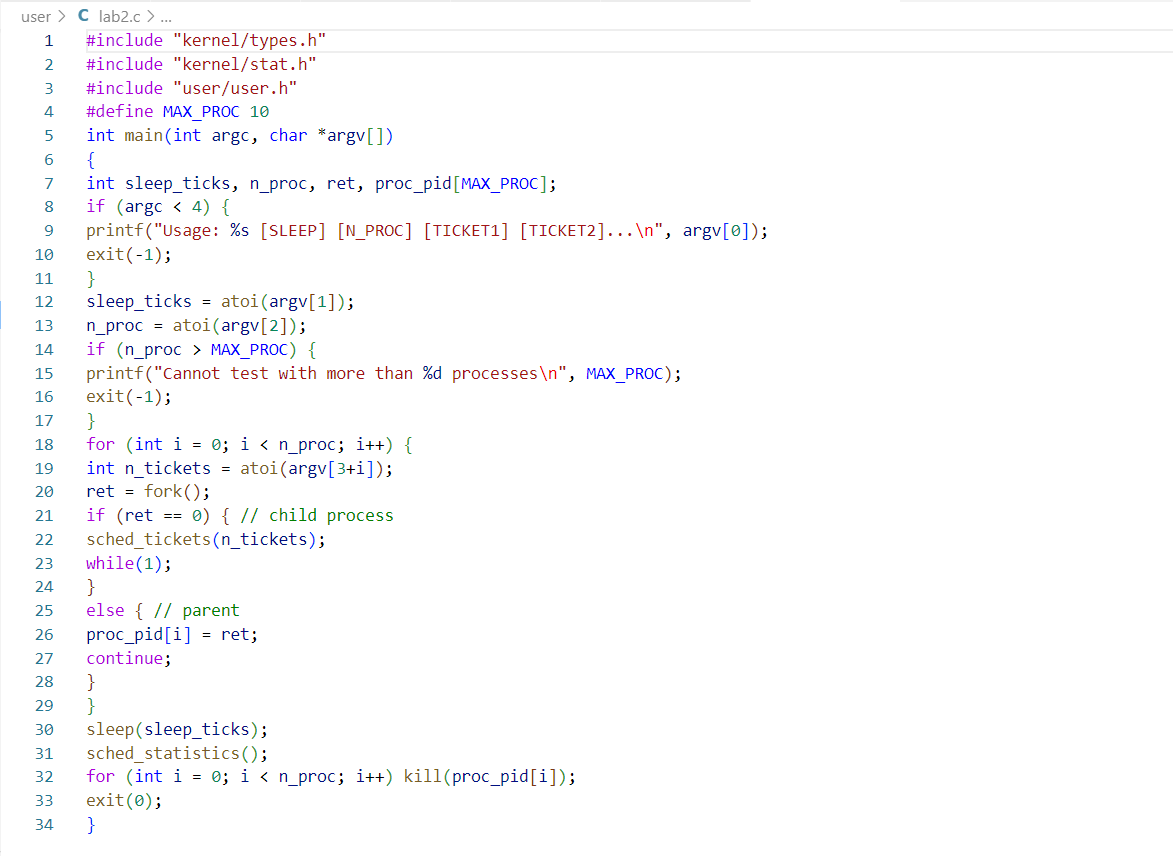
Add lab2 file to make file :



Adding preprocessor directives for switching between different schedulers through command line:

****

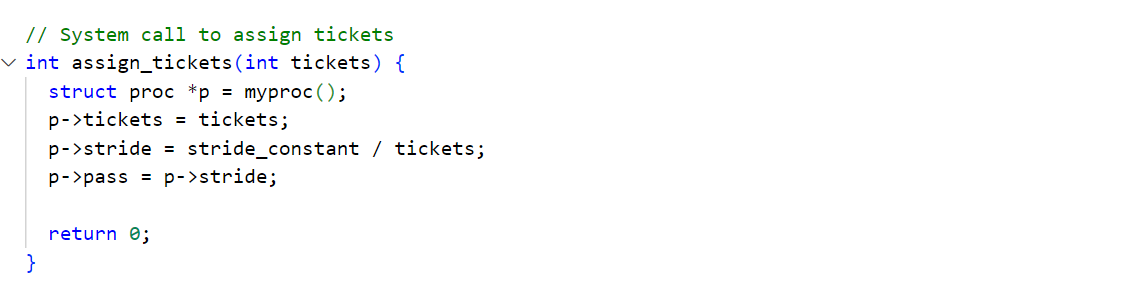
**user/lab2.c:**

****

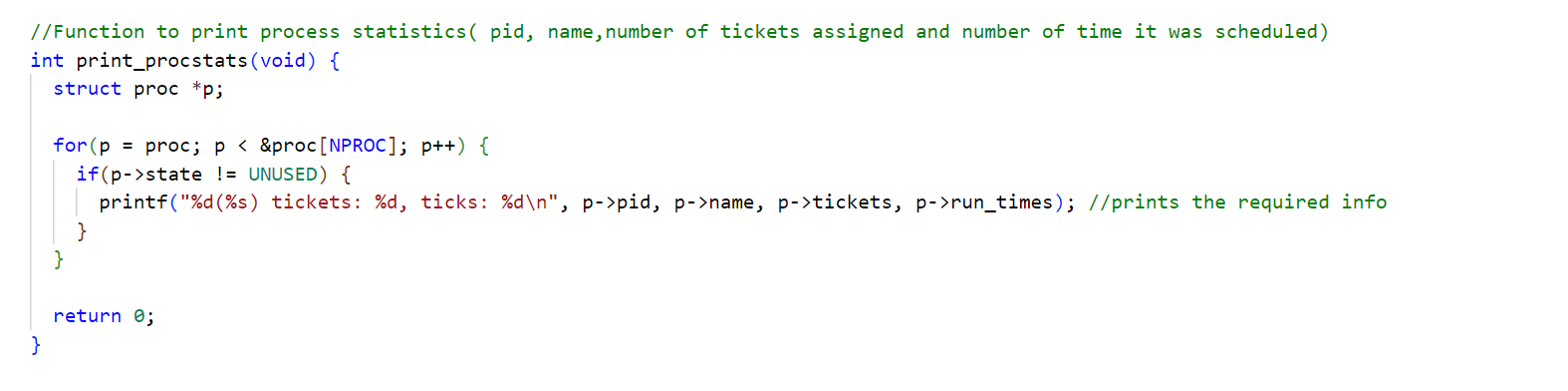
**kernel/proc.c**

Major changes were made to this file. Since, at this location, the scheduler's primary functions are accomplished.

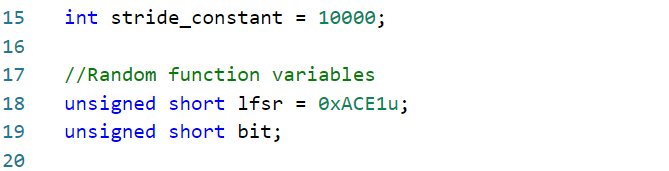
Kernel function to assign the value of tickets.



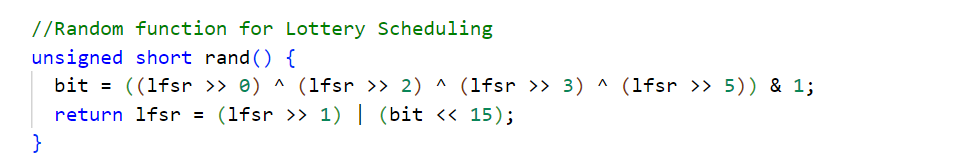
This function is used to print the process statistics.



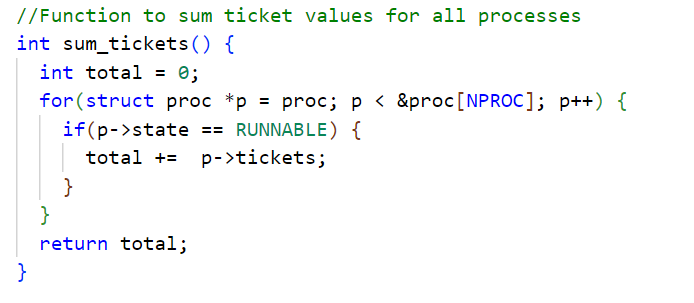
Variable declaration and initializations for Stride scheduling and for Lottery Random function



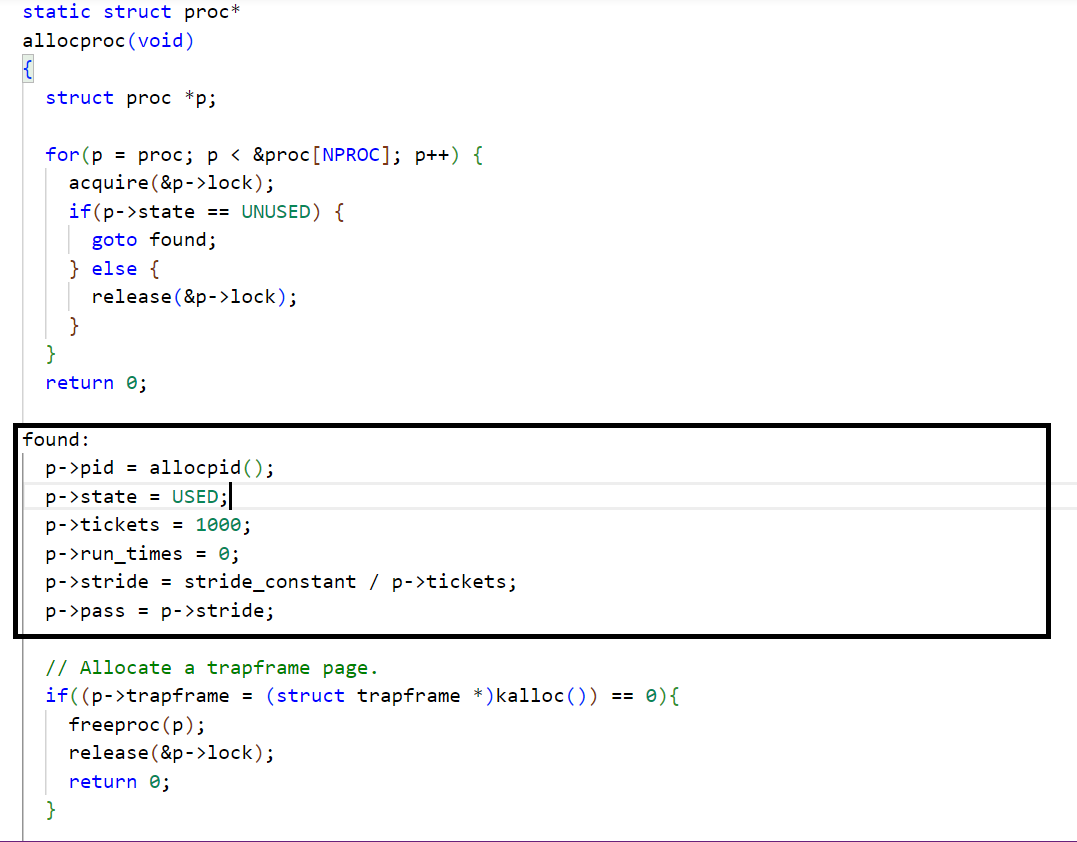
Random function given -



Function to find the total number of tickets that were assigned across all processes.

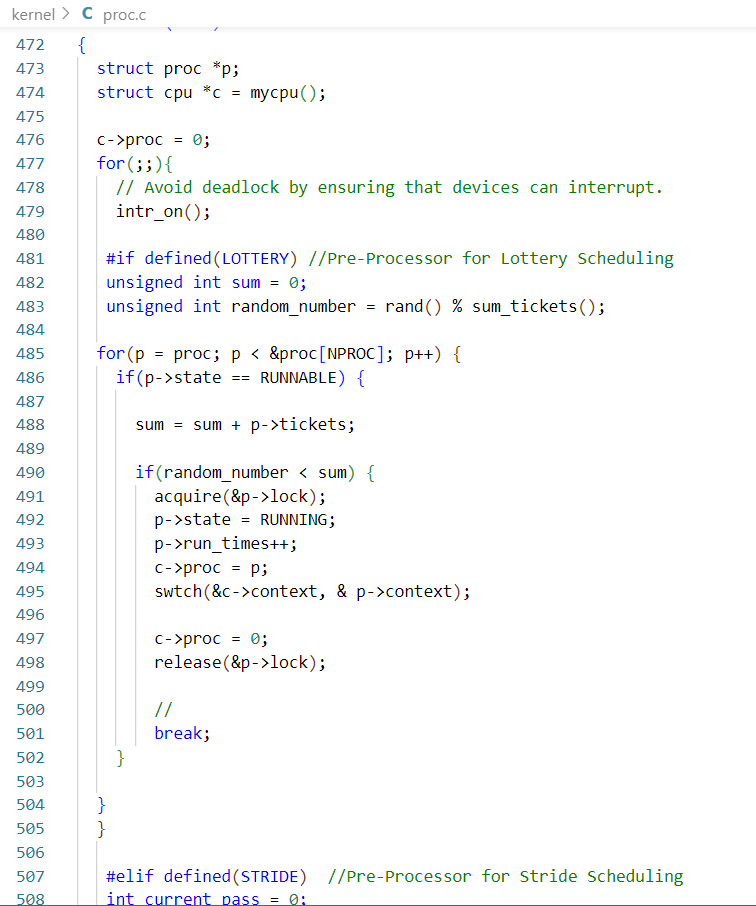


Initialize new per-process variables required for new scheduling algorithms:

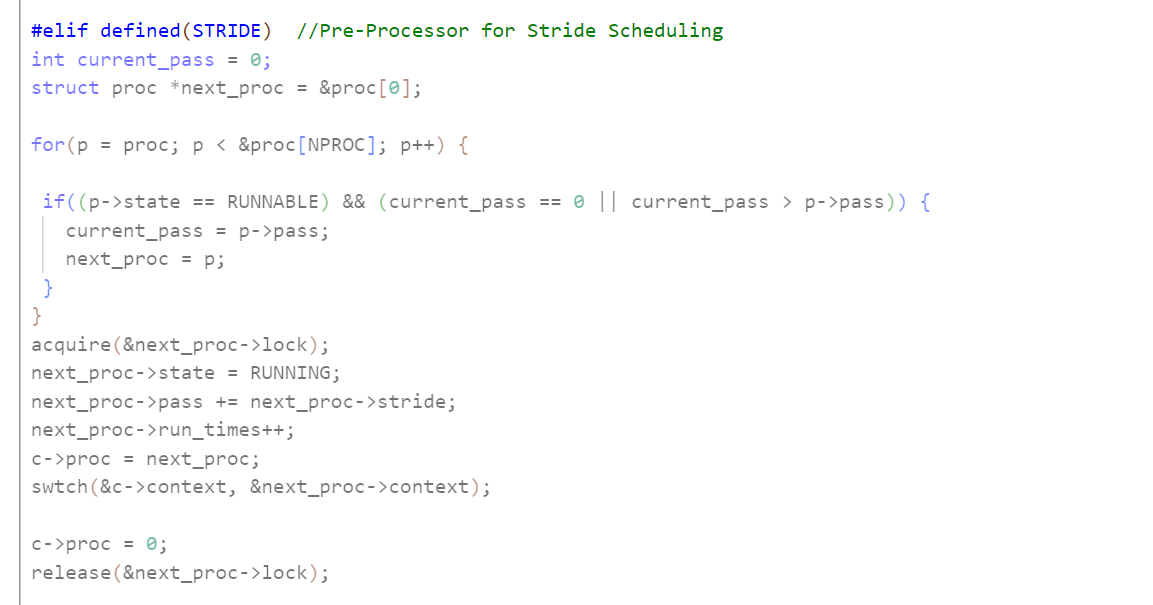


The command in the Makefile controls the scheduler options. It will select the Round Robin scheduler when no option is specified in the command line. It will select the lottery scheduler for the LOTTERY option in the “make qemu” command. It will select the stride scheduler for the STRIDE option in the “make qemu” command.

Lottery Scheduler - Probabilistic scheduler which generates a random number and selects the process which holds the ticket equal to the random number. To find the process holding the lottery winning ticket, we iterate through the RUNNABLE process queue and keep summing the tickets for each iterated process. When ticket sum becomes greater than the random number, the process pointed to in for loop is selected to run.

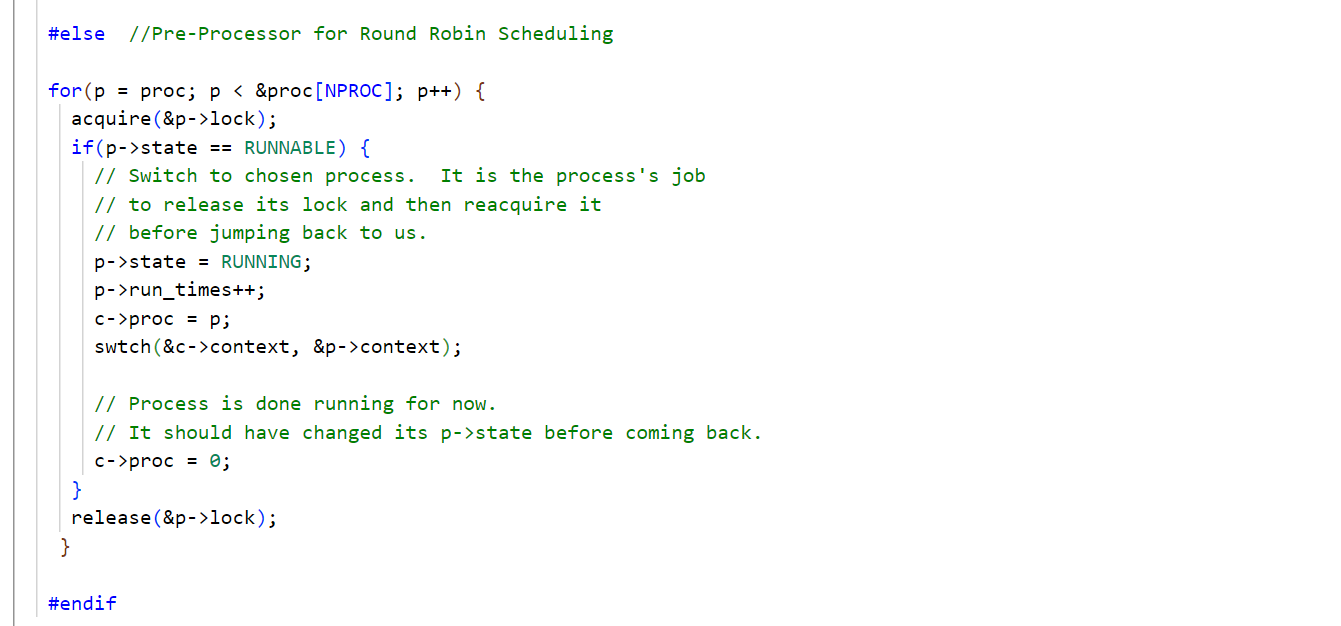


Stride Scheduler - Deterministic scheduler which keeps the pass count of each process and schedules the process with the least pass count. Once a process is scheduled, its pass count is incremented by its stride value. Each process stride value is calculated according to the tickets allocated to the particular process.



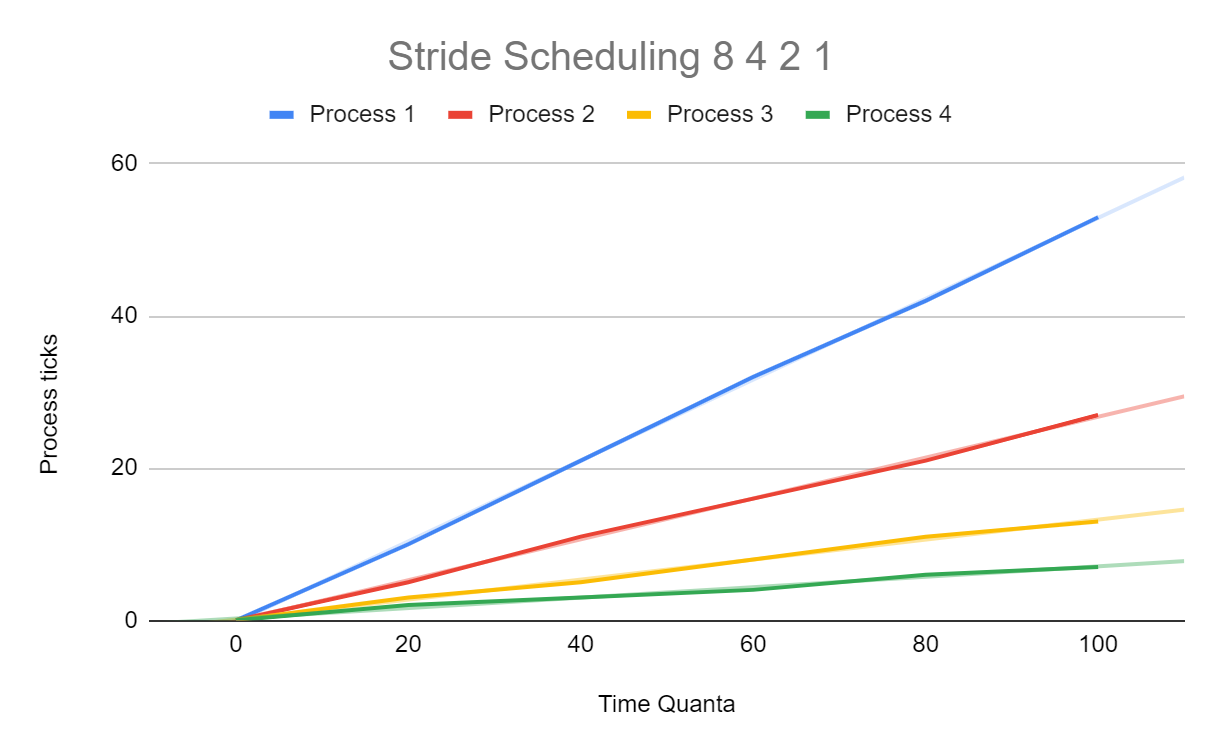
Round Robin Scheduler -

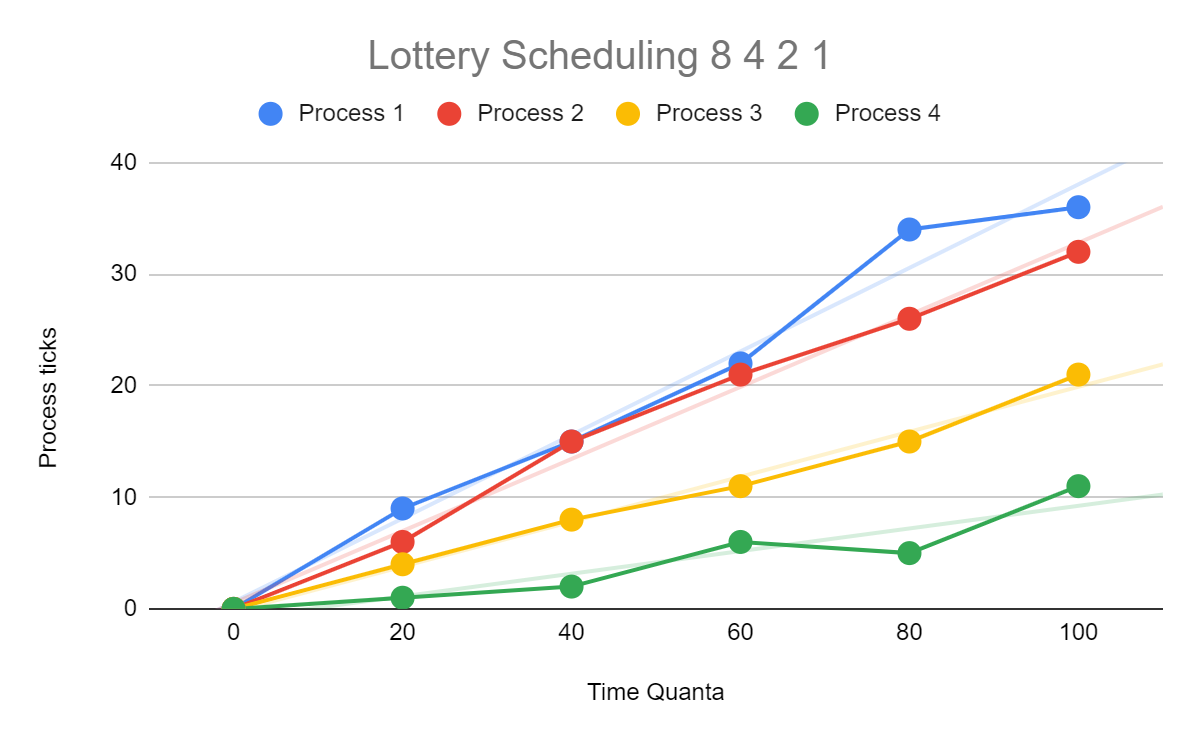
Next process is scheduled for next time slice in a sequential manner, while iterating through the RUNNABLE process queue.



**D.Experiments**

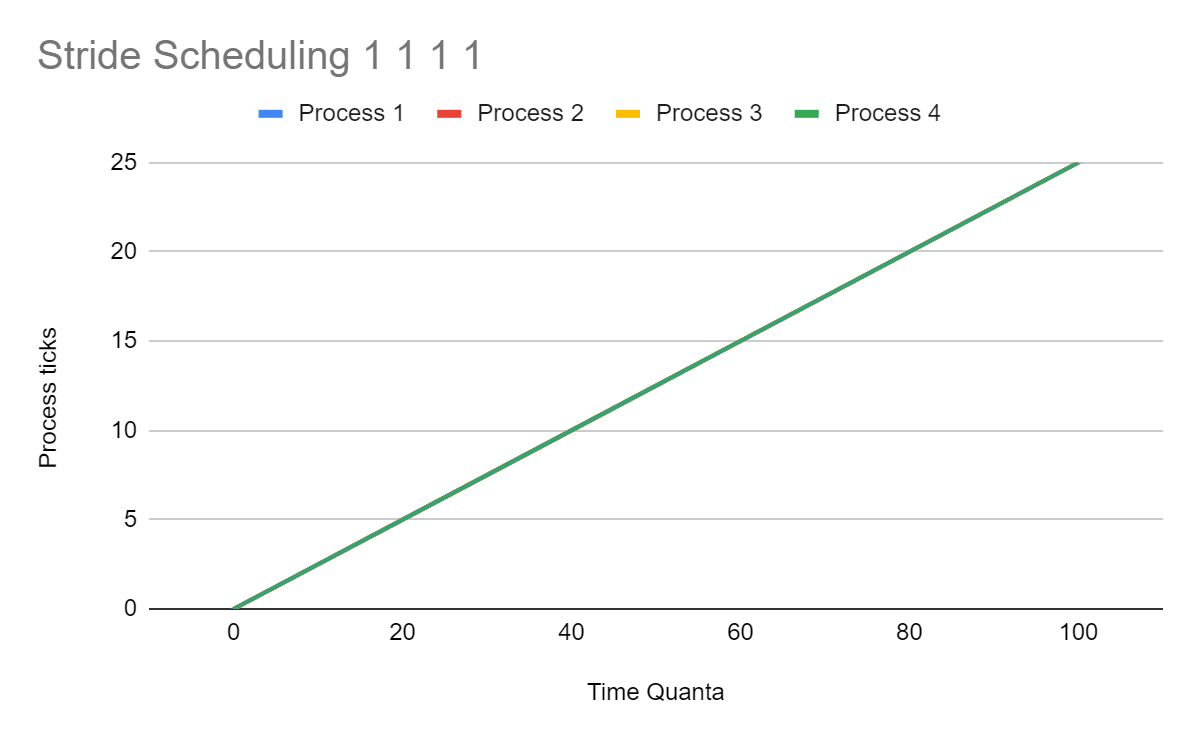
**Experiment 1:** Assigning tickets 8, 4, 2, 1 to 4 processes for Stride and Lottery scheduling:

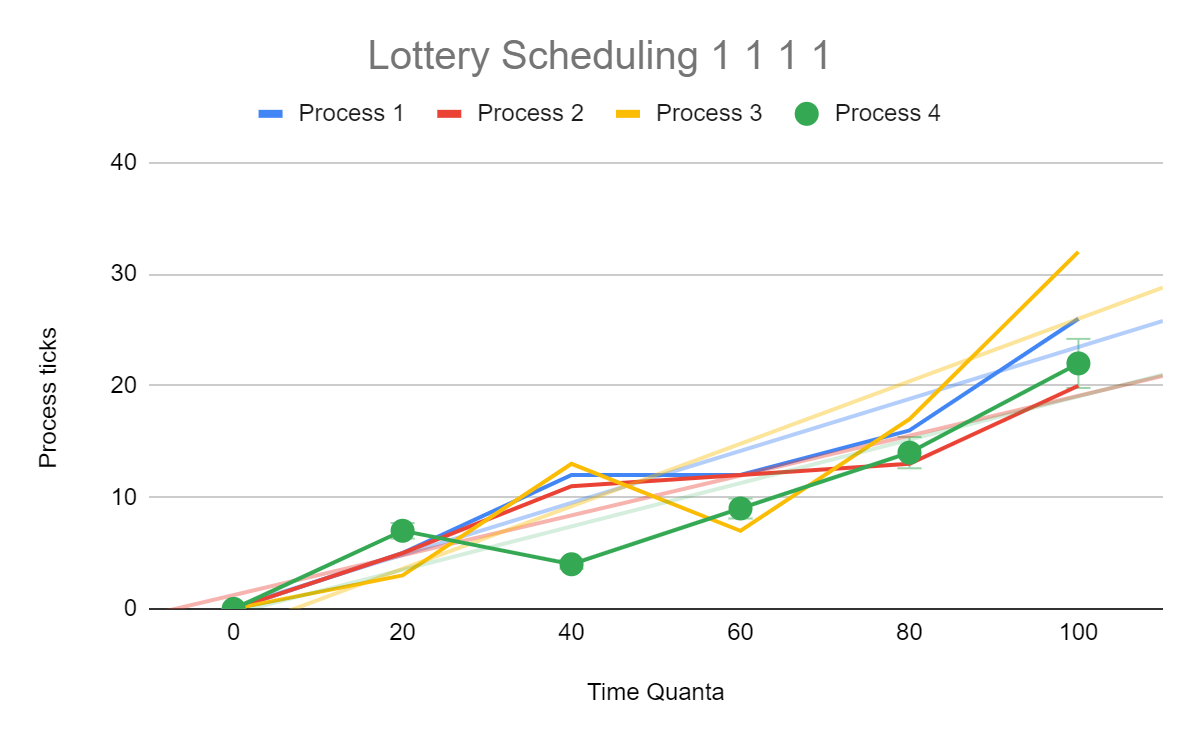
****

****

The above 2 graphs for Stride and Lottery scheduling show us the scheduling pattern for 4 processes respectively to which ticket values of 8, 4, 2, 1 are assigned. The faint lines of the same color show us the trend line / ideal allocations for each process in both schedulers. As we can see from above graphs, both Lottery and Stride algorithms schedule the processes in proportion with the different ticket allocations, but Stride follows the ticket allocation almost perfectly, whereas Lottery scheduling does follow the ticket allocation proportion, but with variations due to randomness.

**Experiment 2:** Assigning 1 ticket to each of the 4 processes in Lottery and Stride scheduling:





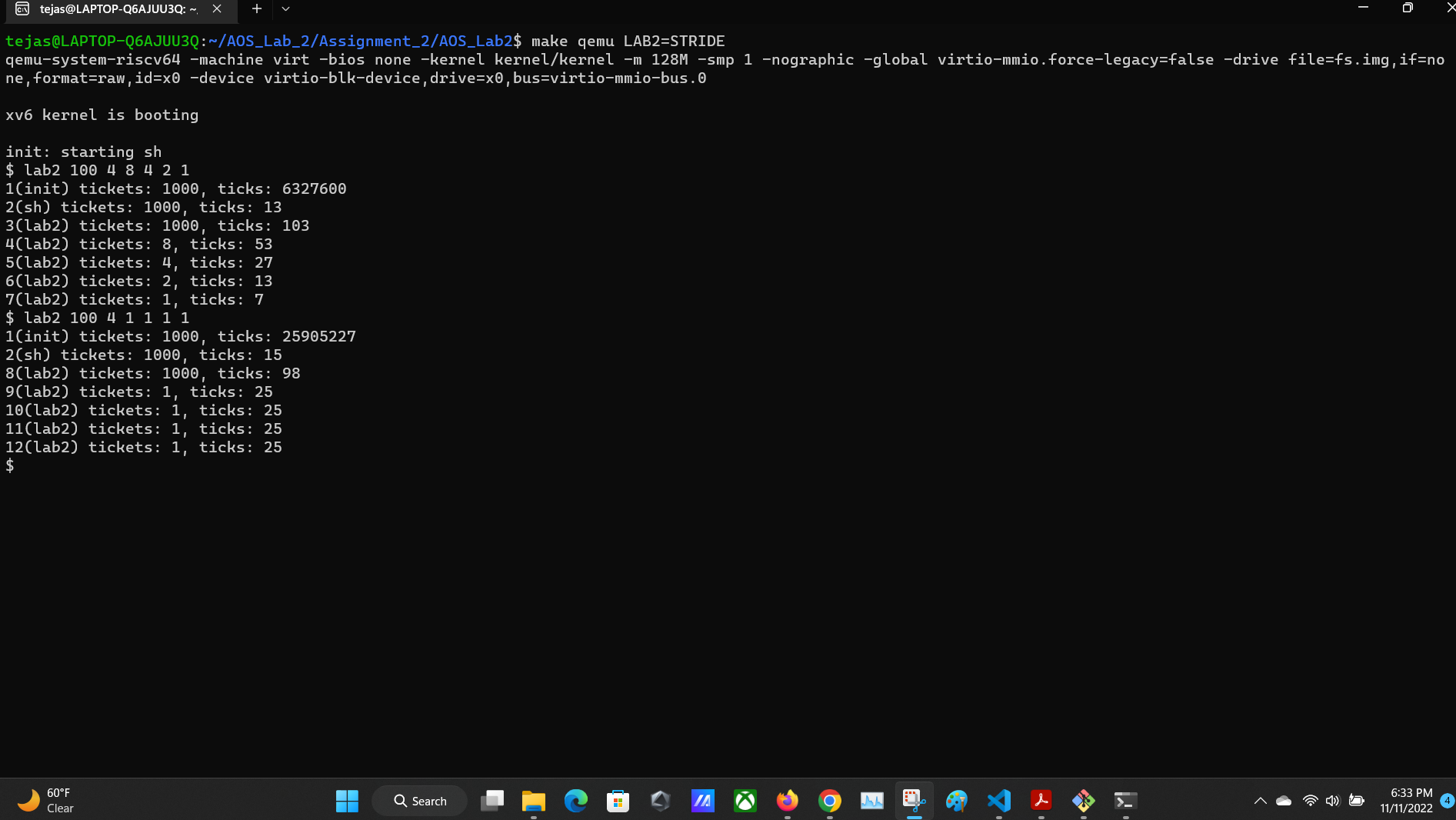
The 2 graphs above for Stride and Lottery scheduling show us the scheduling pattern for 4 processes with equal ticket allocation. For Stride scheduling, we can see only one line because all the lines for all the processes overlap, which means every process has the same allocation pattern due to equal number of tickets. On the other hand, in Lottery scheduling, even though all processes have the same number of tickets, we can see scheduling variation due to randomness of the algorithm.

From the above experiments, we can conclude that Lottery scheduling is probabilistically fair, but exhibits randomness in scheduling the processes, hence it is not deterministic. On the other hand, Stride scheduling is deterministically fair, and does not schedule processes randomly.

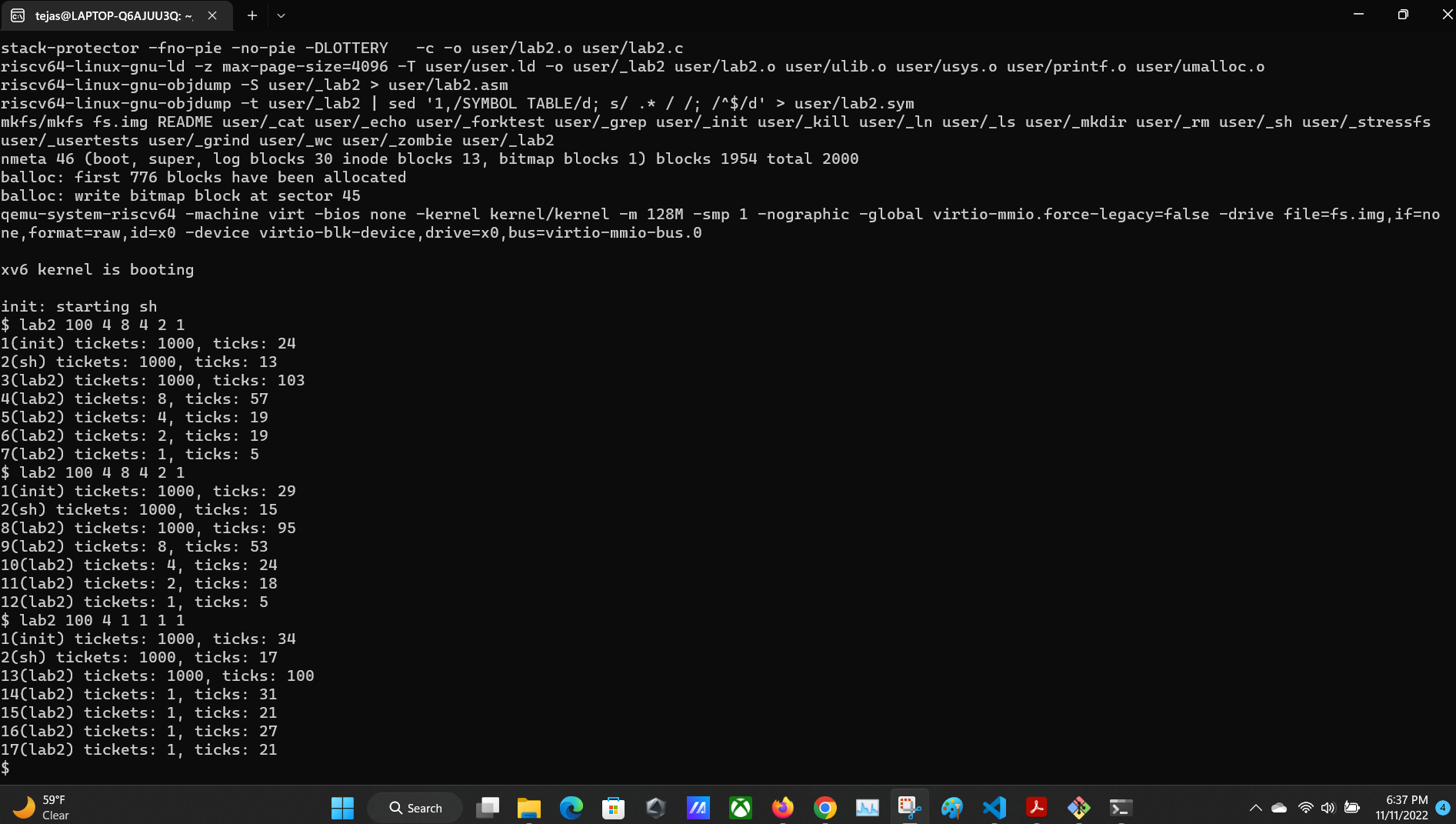
As we can observe, since Stride scheduler is deterministically fair, actual process scheduling coincides with the ideal scheduling trend almost perfectly. In contrast to Stride scheduling, we can see that the Lottery scheduling pattern has a lot of variance between actual allocation and ideal allocation.

**Console output(different execution from graph):**

Stride:

****

Lottery:



**E.Contributions of Each Member**

| **S.No** | **Name** | **Contributions** |
| --- | --- | --- |
| 1) | Tejas Milind Deshpande | * Created Git repository for collaboration. * Implemented Lottery scheduling. * Implemented pre-processor directives. * Integrated the code for peers. * Contributed in the graph plotting, report and video creation. * Reviewed the code of my peers and gave suggestions for improvements. |
| 2) | Nunna Lakshmi Saranya | * Implemented Stride scheduling. * Contributed in the graph plotting, report and video creation. * Studied xv6 scheduler and lottery scheduling. * Reviewed the code of my peers and gave pointers. |
| 3) | Satya sri Nandan Paritala | * Created the two System Calls i.e sched\_tickets and sched\_statistics * Reviewed the code of my peers * Helped in graph plotting,report and video creation |