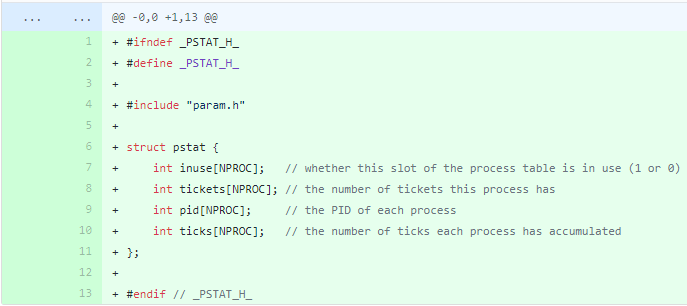
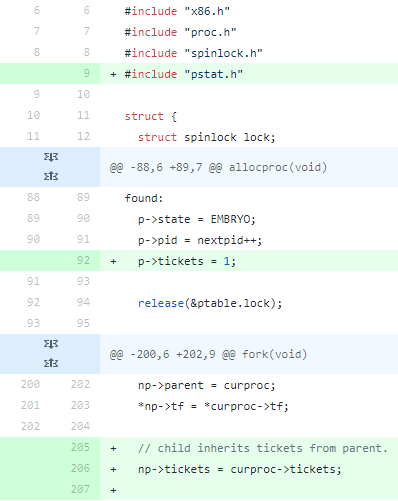
1. pstat.h



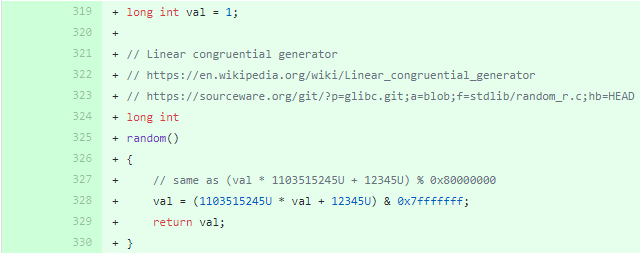
Header file for the relief for recording the status of the current process.

1. proc.c



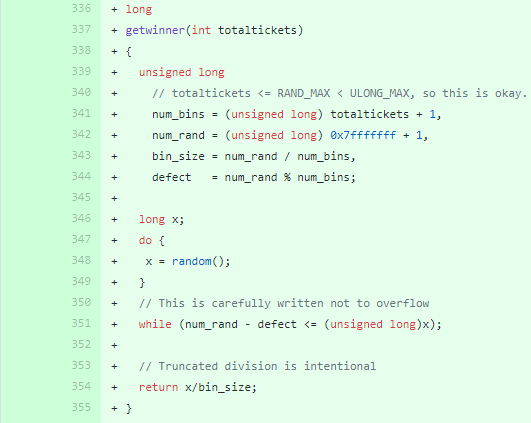
First, add include pstat.h and set tickets.

* 1. random

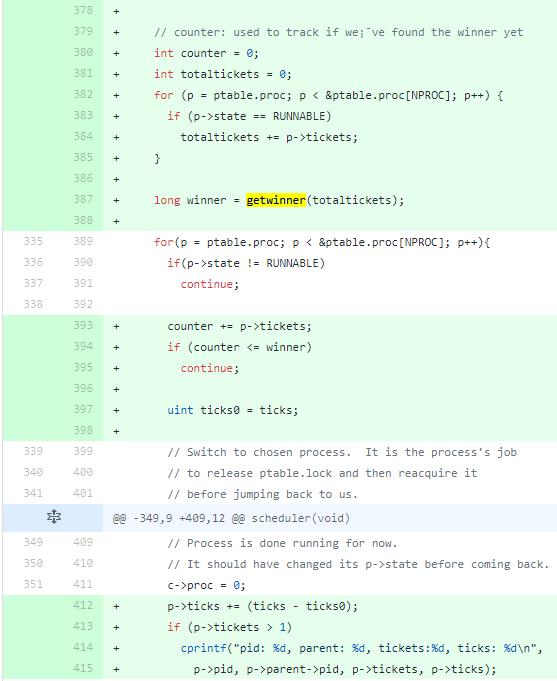


A function that generates random values.

* 1. getwinner

Implement the getwinner function to gry winner.

Generate random x based on totaltikets.



Add the following to the scheduler:

Set the counter and totaltikcet.

And use getwinner function to save winner.

Then repeat NPROC, i.e. the total number of processes, while:

1. If the status of the current process is RUNNABLE, it will be moved to the next process.

2. Add the tickets of the current process to the controller, and if the counter is less than the winner, move on to the next process.

3. Correct the ticks in the current process and add the following to the current pid, parent, tickets, ticksscheduler if the ticks are 1 or more.

Set the counter and totaltikcet.

And use getwinner function to save winner.

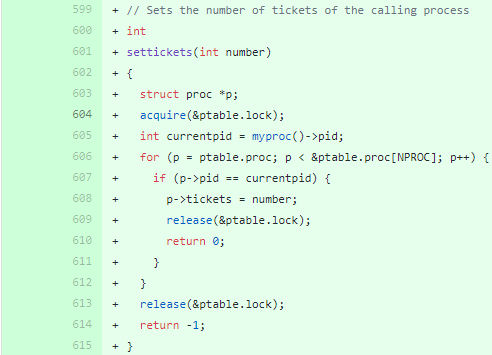
Then repeat NPROC, i.e. the total number of processes, while:

1. If the status of the current process is RUNNABLE, it will be moved to the next process.

2. Add the tickets of the current process to the controller, and if the counter is less than the winner, move on to the next process.

3. Correct the ticks in the current process and output the current pid, parent, tickets, and teaks if the ticks are 1 or higher.

* 1. Settickets



A function that gives the process a ticket, which will be used through the lottery\_sched command.

The contents of the function as follows :

1. Put a lock on the ptable to be used.

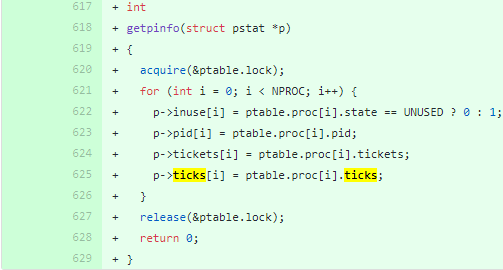
2. Import and set the current pid of myproc.

3. Extract and repeat the process by the maximum number of processes in the NPROC, i.e. the system.

4. If the pid of the process during iteration is the same as the current pid, set the tickets of the process to the number entered as the factor of setcits, then unlock the ptable, and return 0 to mean successful.

5. Unlock the lock of the ptable and return -1 if all processes are turned without the same pid.

* 1. Getpinfo



As a function to show the status of the current process, it will be used through the ps command.

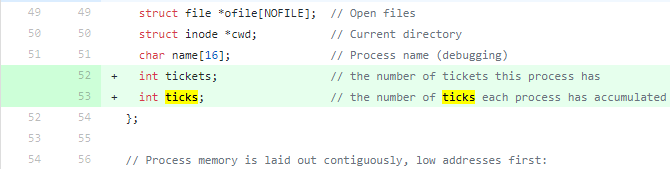
The contents of the function as follows :

1. Put a lock on the ptable that you want to use first.

2. Extract and repeat the process by the maximum number of processes in the NPROC, or system, and record the contents of the ptable on the pstat structure.

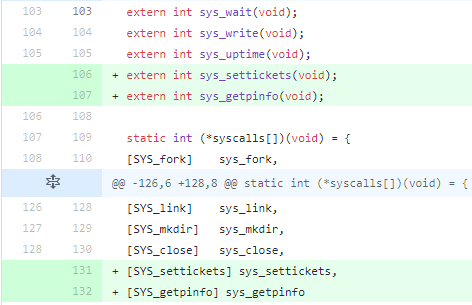
3. Loosen the lock of the ptable when finished.

1. proc.h



Add a member to record the ticks and ticks of the process to the structure proc.

1. syscall.c

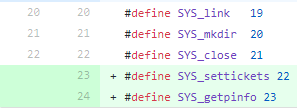


[SYS\_settlecits] sys\_setcits, [SYS\_getpinfo] sys\_getpinfo and external int sys\_setcits(void),

Add external int sys\_getpinfo(void).

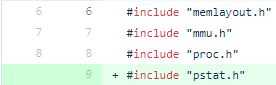
It is declared external so that references can be made in other processes and files.

1. syscall.h



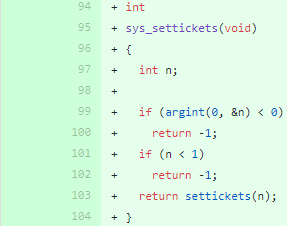
Add sys-call vector with #define SYS\_setticets 22, #define SYS\_getpinfo 23.

1. sysproc.c



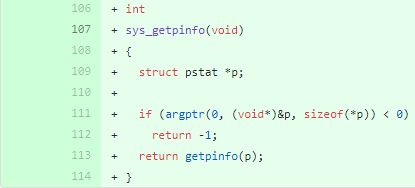
First, add include pstat.h and proc.h.

* 1. sys\_settickets



Passing the 0th aggregate to n via argint, if the result is less than zero, returns -1 because it is a failure.

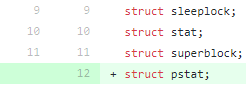
* 1. sys\_getpinfo



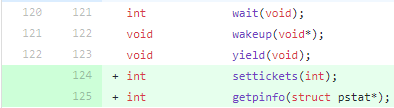
First, add structure pstat.

Then, pass the corresponding pstat structure to the 0th aggregate via argptr, which returns -1 because if the result is less than zero, it is a failure.

1. defs.h



First, add structure pstat.

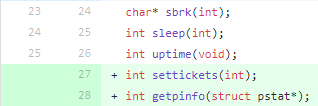


Add int sets(int) and int getpinfo(construct pstat\*) under proc.c.

1. user.h

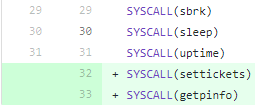


First, add structure pstat.



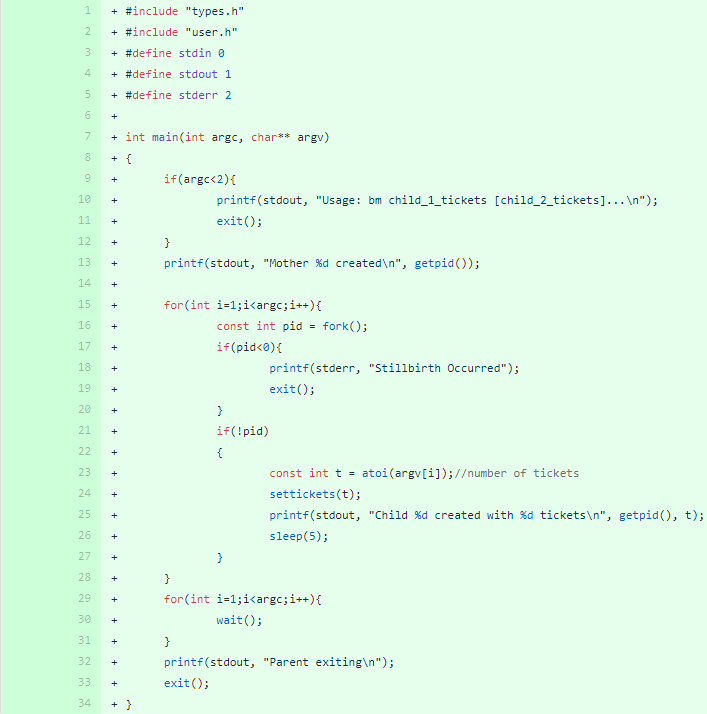
Add int sets(int) and int getpinfo(construct pstat\*) under system call.

1. usys.S



Add SYSCALL(setcits) and SYSCALL(getpinfo).

1. lottery\_sched.c



This is a file to create a command using settickets.

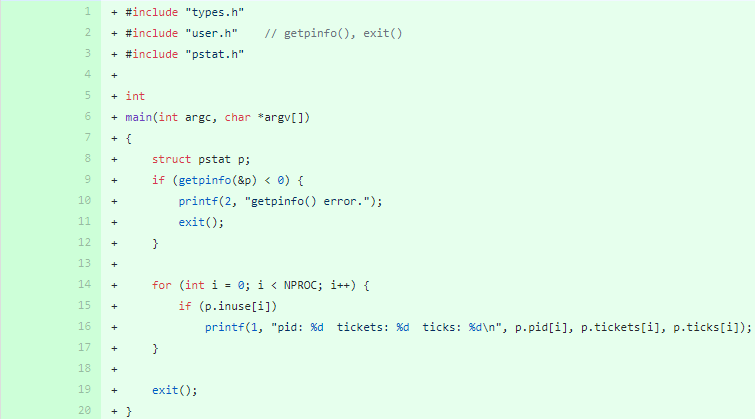
The contents of the function as follows :

1. the structure of the input is lottery\_sched children\_1\_tickets [child\_2\_tickets]. Therefore, check if it is correct.

2. If correct, run setttickets() with arguments.

3. Show pid and mother according to execution result.

1. ps.c



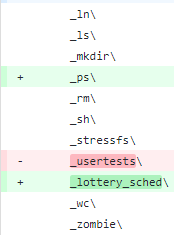
This is a file to create a command using getpinfo.

The contents of the function as follows :

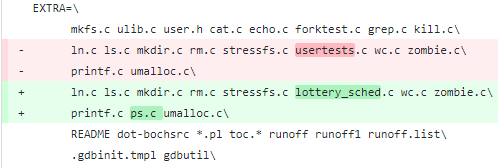
1. Record the status of the processes in the pstat structure variable through getpinfo().

2. Print out pid, ticks, and ticks while touring the entire process.

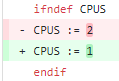
1. Makefile



Add \_ps and \_lottery\_sched to UPROGS.



Add ps.c and lottery\_shced.c to the EXTRA.



It can also be more intuitive to show progress by adjusting the number of cpu to one.