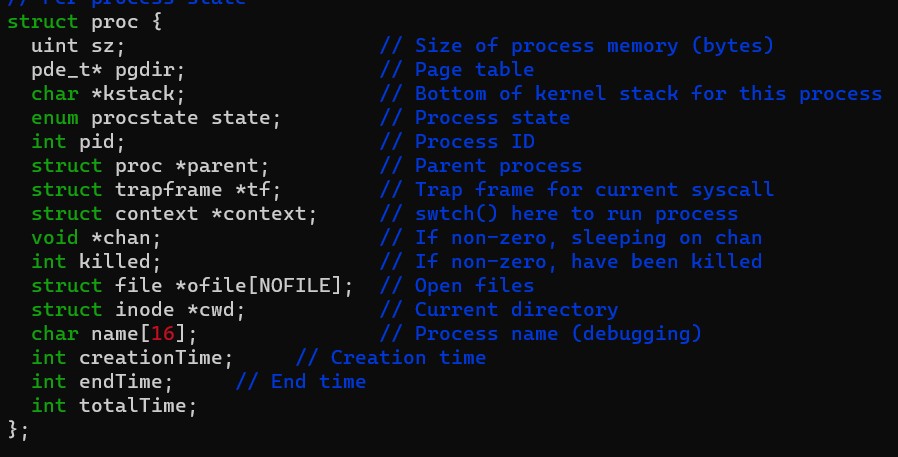
**OPERATING SYSTEM ASSIGNMENT -2**

**Part-1** Getting process statistics and modifying system call in xv6- 300  
points

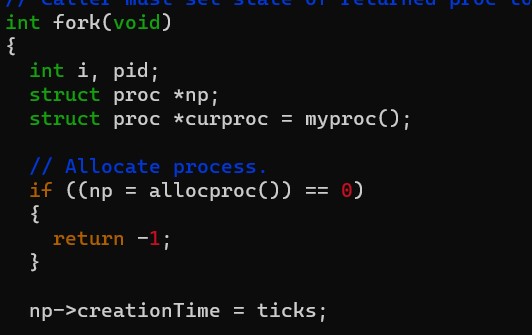
In this task we will get the creation time, end time, total time for the uniq and head operations.

We use the system calls of uniq and head operations which we have created for the assignment 1.

Modify **proc** structure: Open **proc.h** in the xv6 source code and add the following fields to the **struct proc**



Whenever a new process is created, you should update its **creation\_time** field. You can do this in the **fork()** system call. Open **proc.c** and locate the **fork()** function. Update it to set the **creation\_time** field for the child process.

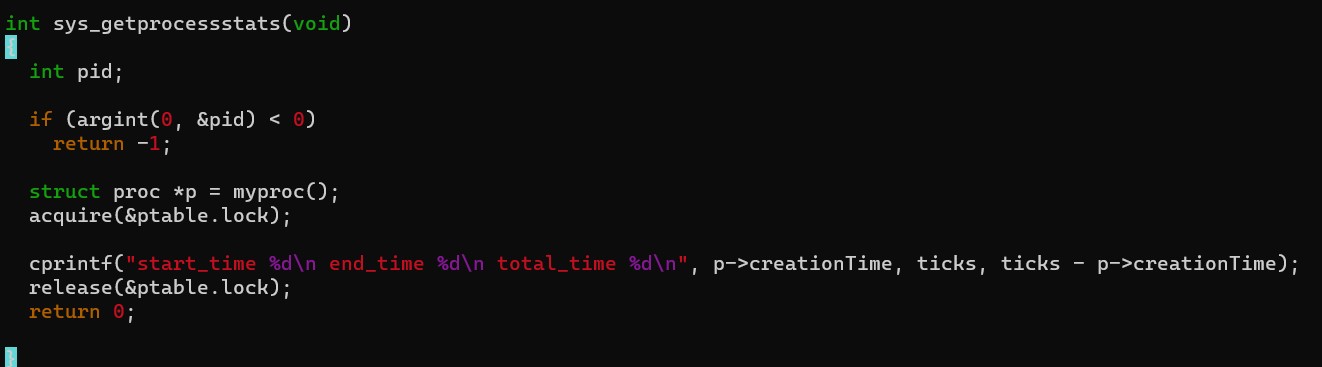


To track the end time of a process, you should update the **exit()** system call in **proc.c**. When a process exits, set the **end\_time** field for that process.



By using start and end time ewe can calculate the total time taken.

Create a new system call:



Modify syscall.c, syscall.h, user.h, usys.S to handle the system call

In syscall.c

C:\Users\gsuch\OneDrive\Desktop\project2\part1\screenshots\Screenshot 2023-09-29 141351.jpg

C:\Users\gsuch\OneDrive\Desktop\project2\part1\screenshots\Screenshot 2023-09-29 141428.jpg

In syscall.h

C:\Users\gsuch\OneDrive\Desktop\project2\part1\screenshots\Screenshot 2023-09-29 135254.png

In user.h

C:\Users\gsuch\OneDrive\Desktop\project2\part1\screenshots\Screenshot 2023-09-29 141706.jpg

In usys.S

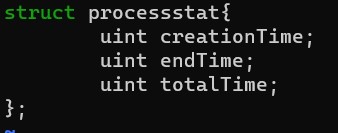
C:\Users\gsuch\OneDrive\Desktop\project2\part1\screenshots\Screenshot 2023-09-29 141744.jpg

Write a **test.c** program in user space that calls your new system call for the **uniq** and **head** processes

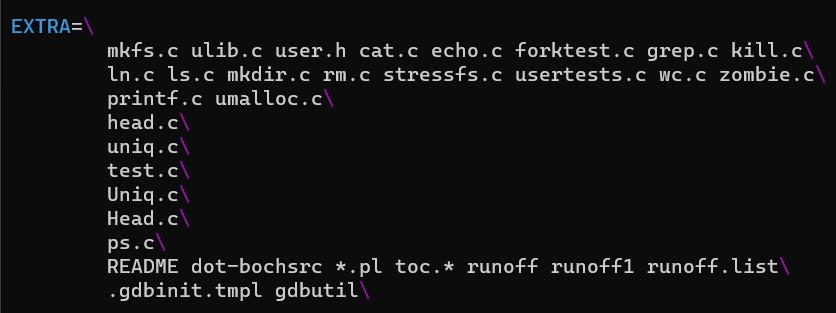


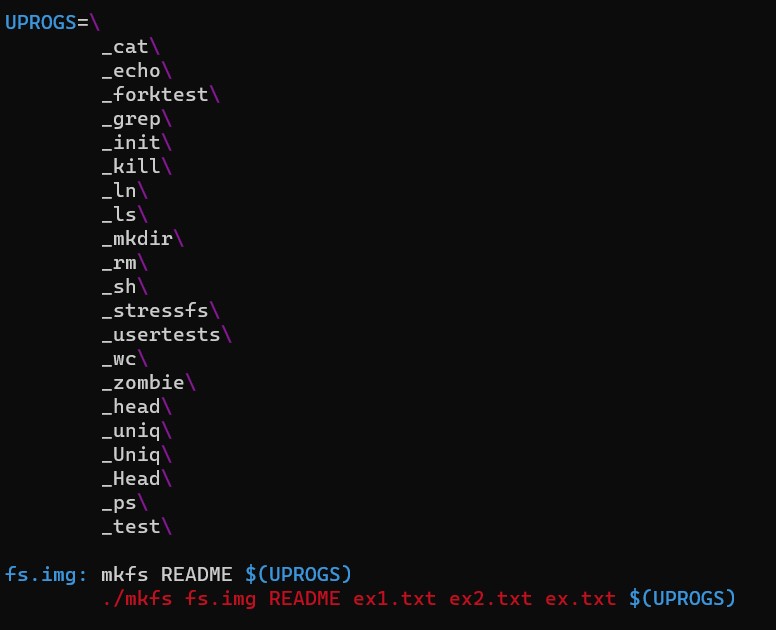


Where we have are using struct processstat which we have declared in types.h

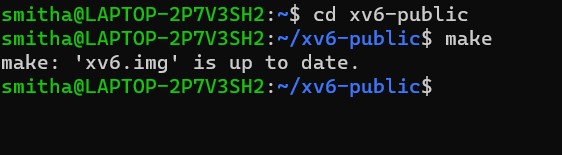


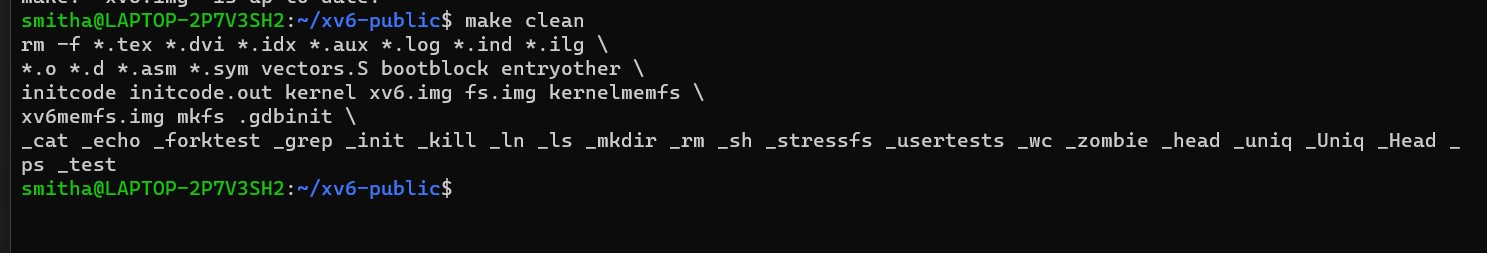
Modify the Makefile





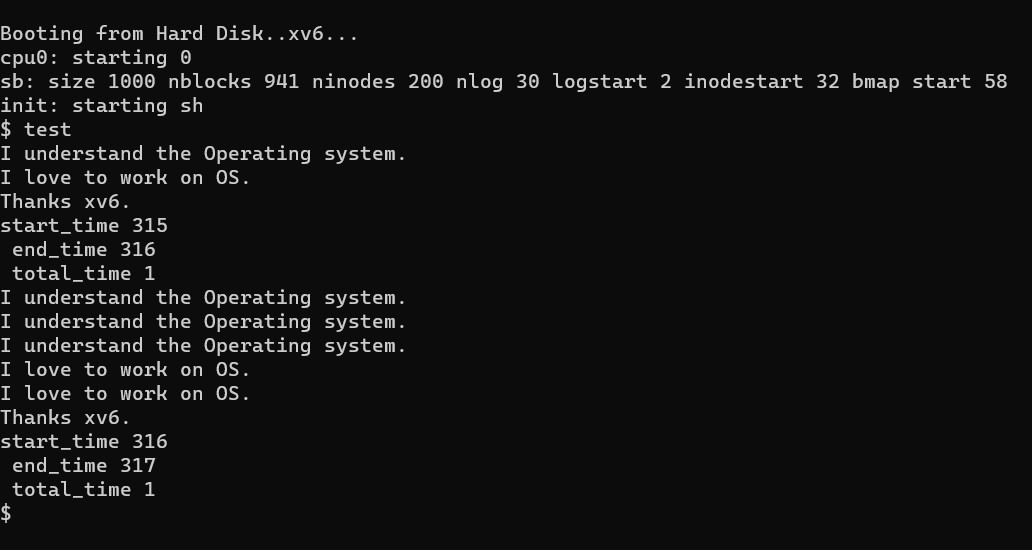
Execute the test.c





C:\Users\gsuch\OneDrive\Desktop\project2\part1\screenshots\Screenshot 2023-09-29 135505.jpg

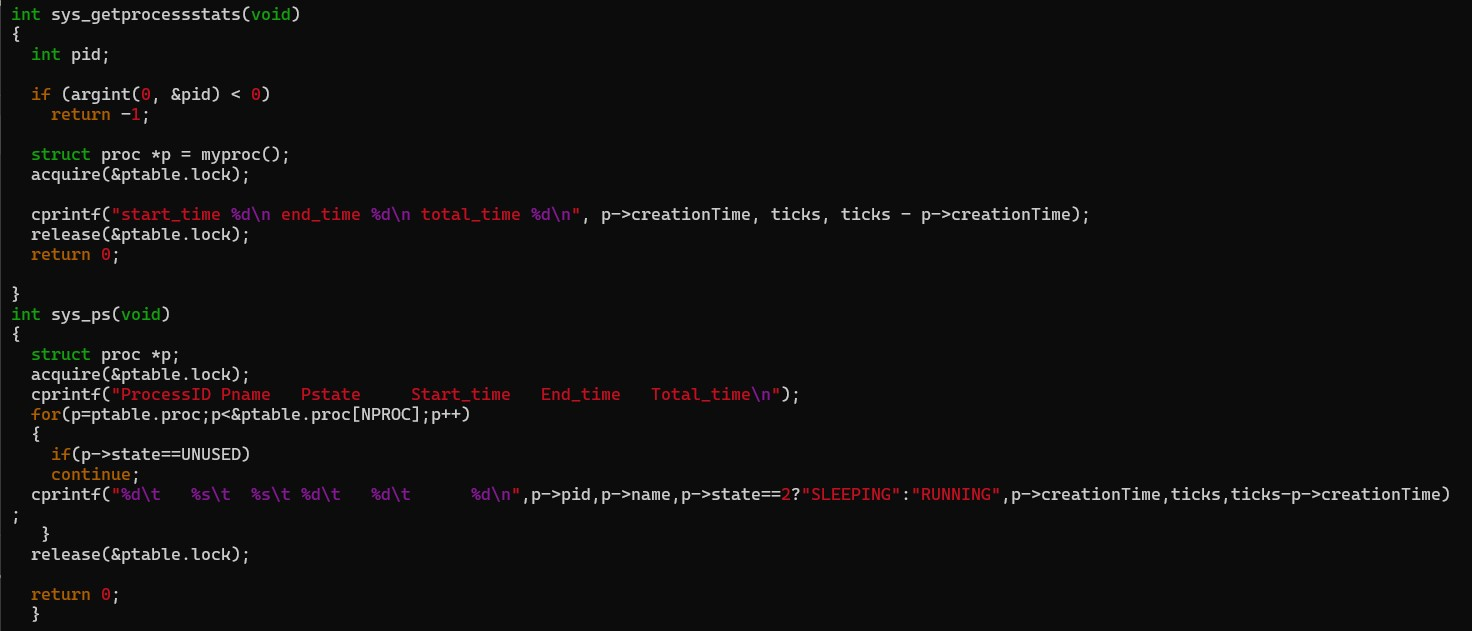
**Output:**



**TASK 2:**

Implementing ps on xv6 - 300 points.  
Once part 1 is complete, you must create a ps command in xv6, which provides information about the process. In your custom ps, you will display PID number, process status (running, zombie, wait etc), start time, total time, and process name.

We need to write a system call which gives information about the process in proc.c.



Modify syscall.c, syacall.h, user.h, usys.S to handle the system call

In syscall.c

C:\Users\gsuch\OneDrive\Desktop\project2\part2\screenshots\Screenshot 2023-09-29 145224.jpg

C:\Users\gsuch\OneDrive\Desktop\project2\part2\screenshots\Screenshot 2023-09-29 145137.jpg

In syscall.h

C:\Users\gsuch\OneDrive\Desktop\project2\part2\screenshots\Screenshot 2023-09-29 145302.jpg

In user.h

C:\Users\gsuch\OneDrive\Desktop\project2\part2\screenshots\Screenshot 2023-09-29 145341.jpg

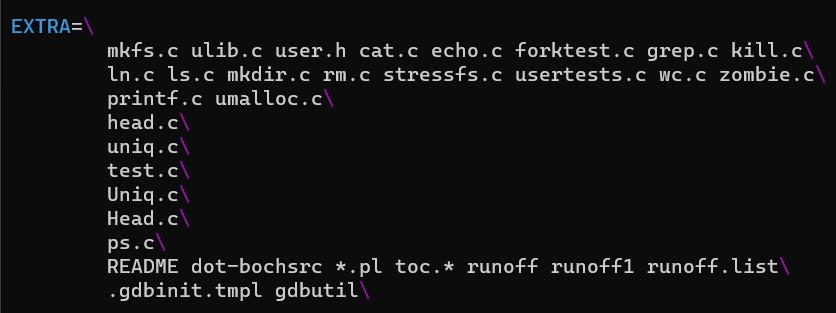
In usys.S

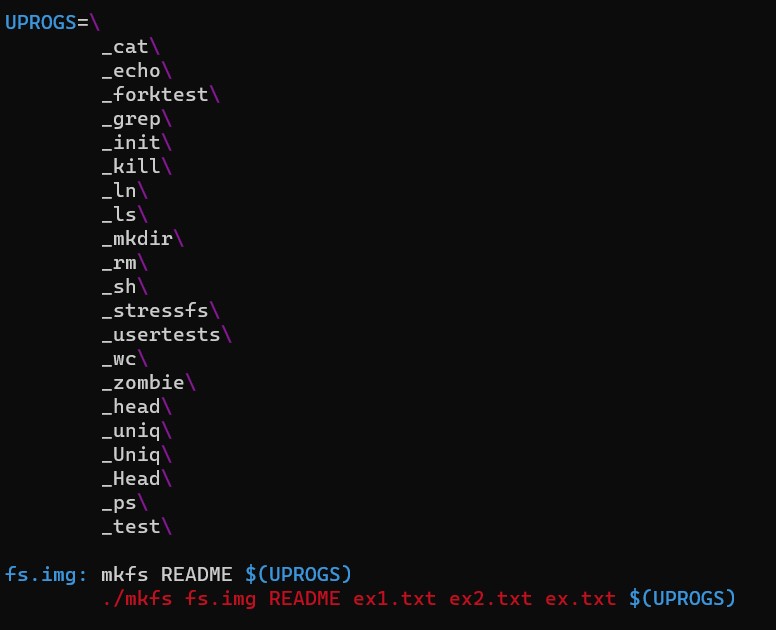
C:\Users\gsuch\OneDrive\Desktop\project2\part2\screenshots\Screenshot 2023-09-29 145416.jpg

We need to write a file in user mode to call the system call

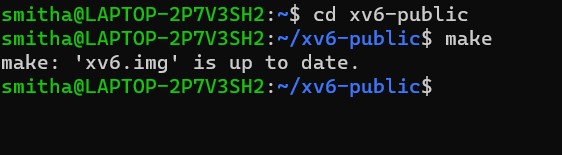


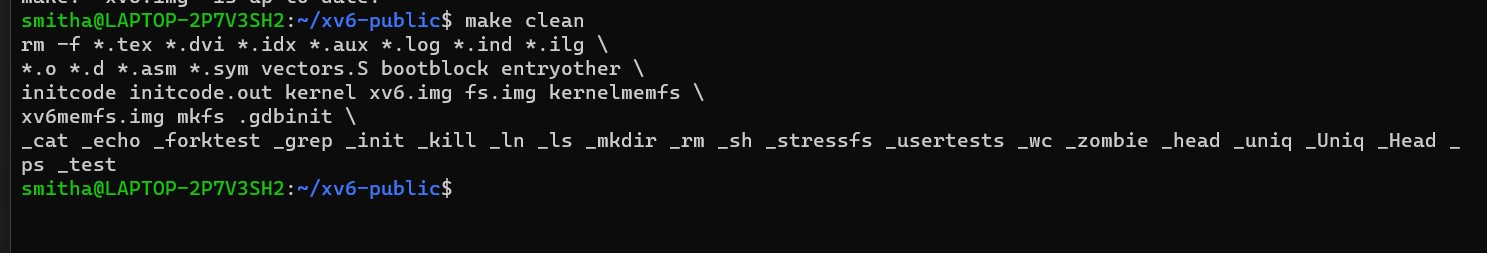
Modify the Makefile





Use make, make clean, make qemu-nox to get the output.





C:\Users\gsuch\OneDrive\Desktop\project2\part2\screenshots\Screenshot 2023-09-29 135505.jpg

**Output:**

