# Operating Systems–2: CS3523
# January 2022

## Programming Assignment 7: Priority Scheduling

Submission Date: 1st May 2022 (Sunday), 9:00 pm

---

You can continue on this assignment using the same xv6 repo where you implemented the system call in Assignment-6 or just start from a fresh one, as you are comfortable.

While we discussed xv6 code in detail in our lectures, there is a book containing internal details of xv6, to understand some deeper insights into xv6. It is located at: https://drive.google.com/file/d/1ujCX6t0T6o9MGR9vrUdsyWQmCkygByn3/view?usp=sharing

## Part-1: System call to assign/change the priority of any process.

Your first task is to implement a system call to assign priority to the current process. Whenever a new process is created, it is assigned a default priority number. However, the process can request the OS to change its priority to a higher or lower value through a system call (**setPriority**). setPriority takes a number as the argument and assigns that value as priority to the current process.

For simplicity, we consider that there can be only 10 priority levels - from 0 to 9. The system call should give an error if we assign a priority outside the allowed range. The default priority is 5.

**Hint**: Where to store the priority value? Add another field to the process state structure so that the priority can be stored during process allocation as well as updated during the system call.

What can be a problem with supporting such a system call in any OS? How could we mitigate this problem? Please write in your report.

## Part-2: Scheduling policy based on priority of processes

Now, we update the default CPU scheduling policy for processes. We consider process priority as a factor to determine scheduling decisions. A higher priority process is scheduled first. If there are multiple processes of the same priority, they should be scheduled in a round-robin manner.

**Hint:** Process scheduling is implemented inside proc.c. What is the default implementation policy in xv6?

There is a problem with such a priority based policy. What is that problem? How can it be mitigated? Please suggest in your report.

**Important**: While you might add print statements in the code for your debugging purposes, the final submission should not contain any extra print statements from the kernel. However, it is perfectly okay to have any print statements in the user code implementation.

# User program for testing

For each of these parts, we need to implement a user program to test out the functioning of the features. What would such a user program look like? We should be able to test various situations to test our implementation. Plan well and then start coding for it. The user program must be implemented inside a file named **myPrioritySched.c**
We might use a different user program during evaluation of your submission, hence do thorough testing of your implementation.

# Submission Instructions

Submission is to be done at the appropriate link. Just bundle the modified files as per below instructions.

1. Go inside the xv6 directory
2. make clean
3. tar -zcvf xv6.tar.gz * --exclude .git
4. A small report (report.pdf) with max. 2 pages explaining your implementation in brief, observations for various experiments, and your learning from this assignment.
5. Create a zip file containing xv6.tar.gz and report.pdf and upload it. The zip file should follow the name: Assgn7-<RollNo>.zip

# Grading Policy

1. Part-1 working: 30%
2. Part-2 working: 40%
3. Report: 30%

**Note:** We would run plagiarism checks on the submissions and copy cases would be appropriately dealt with. If needed, we might conduct a viva to confirm the same.