CSCI 5573 Homework Assignment 1 Spring 2026

Due Date: 5 pm, February 11 (Wed), 2026

1) Disclaimer

This homework is based on our textbook, Operating Systems: Three Easy Pieces, written by Remzi and Andrea Arpaci-Dusseau at the University of Wisconsin.

2) Goal

In this homework, you will familiarize yourself with the process management APIs we learned.

3) Requirement

Write all codes with either C or C++, and you must test your code on the CSE department Linux cluster, csegrid.ucdenver.pvt.

4) What to submit & how to submit

a) Answer all questions and add print output for each question.

b) All source codes in tar format, DO NOT INCLUDE *.O and executable image files.

- Please use the following convention when you create a tar file

  ①. Letters of your last name and the last 4 digits of your student ID

  ②. e.g., If your name is "James Bond" and your ID is 201812345, then your tar file name is "bon2345.tar".

  ③. If you want to know more about the "tar" command, type "man tar" at Linux prompt

c) Upload all required materials to the class Canvas.

5) Questions

1. Write a program that calls fork(). Before calling fork(), have the main process access a variable (e.g., x) and set its value to something (e.g., 100). What value is the variable in the child process? What happens to the variable when both the child and parent change the value of x?

2. Write a program that opens a file (with the open() system call) and then calls fork() to create a new process. Can both the child and parent access the file descriptor returned by open()? What happens when they are writing to the file concurrently, i.e., simultaneously?

3. Write a program that calls fork() and then calls some form of exec() to run the program /bin/ls. See if you can try all of the variants of exec(), including (on Linux) execl(), execle(), execlp(), execv(), execvp(), and execvpe(). Why do you think there are so many variants of the same basic call?

4. Now write a program that uses wait() to wait for the child process to finish in the parent. What does wait() return? What happens if you use wait() in the child?

5. Write a slight modification of the previous program, this time using waitpid() instead of wait(). When would waitpid() be useful?

6. Write a program that creates a child process, and then in the child closes standard output (STDOUT FILENO). What happens if the child calls printf() to print some output after closing the descriptor?

7. Write a program that creates two children and connects the standard output of one to the standard input of the other using the pipe() system call.