CSC 456 Operating Systems
Spring 2021
Krebsbach
Assignment #1 - *Chapter 3* 15 points
DUE: 11:00 AM (CST) Tue Feb 2nd

You should complete the D2L QUIZ to record your answers.

The quiz will have a 15 min time window to record your answers

1. **Assume the code in FIG 1 (below) is submitted for execution.** Under what circumstance would the line of code **printf("** HERE ** \n");** in FIG 1 (next page) be reached and printed out.

   note: you may have to do a little research on execlp()

2. How many new processes are created by the program in FIG 2?

Processes = _____

3. Using the program in FIG 3, identify the value printed at line **A, B, C**, and **D**.

   (**Assume that the "actual" pid of the parent is 2222 and child is 1111)**

		Possible values
A - pid	_____	**1111 or 2222 or 0**
B - pid1	_____	**1111 or 2222 or 0**
C - pid	_____	**1111 or 2222 or 0**
D - pid1	_____	**1111 or 2222 or 0**

========== **FIG 1** ==========

```
int main( void)
{
        pid_t   pid , pid1;

        pid = fork();   // fork a child process

        if (pid < 0) // error
        {
                fprintf(stderr, "Fork Failed");
                return 1;
        }

        else if (pid == 0) // child
        {
                execlp("/bin/ls","ls",NULL);
                printf("** HERE ** \n");
        }

        else //parent
        {
                wait(NULL);
                printf("Parent is ending");
        }

        return 0;
}
```

========== **FIG 2** ==========

```
int main(void)
{
 int i;

 for (i = 0; i < 3; i++) // loop 3 times
 {
        fork();
 }

 return 0;
}
```

========== **FIG 3** ==========

```
int main(void)
{
        pid_t   pid , pid1;

        pid = fork();     // fork a child process

        if (pid < 0)      // error
        {
                 fprintf(stderr, "Fork Failed");
                return 1;
        }

        else if (pid == 0)    // child
        {
                pid1 =  getpid();    // returns the PID of the caller process
                printf("pid = %d   \n", pid);     // A
                printf("pid1 = %d  \n", pid1);  // B
        }

        else //parent
        {
                wait(NULL);
                pid1 =  getpid();      // return the PID of the caller process
                printf("pid = %d \n", pid);       // C
                printf("pid1 = %d \n", pid1);    // D
        }

        return 0;
}
```