CSI 402 – Systems Programming – Handout 13.3
An Example Using execlp System Call

Note: The following example uses three system calls, namely, `fork`, `wait` and `execlp`. The child process runs the command `"grep test infile.txt"` using the `execlp` system call. After the `fork` system call, the parent process waits for the child to complete. When you run this program, the output produced by the `"grep test infile.txt"` command from the child will appear first and then the output produced by the parent.

```c
#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <string.h>

int main(void) {
  pid_t  child;
  int    cstatus;  /* Exit status of child. */
  pid_t  c;        /* Pid of child to be returned by wait. */

  if ((child = fork()) == 0) {

     /* Child process.  To begin with, it prints its pid. */
     printf("Child: PID of Child = %ld\n", (long) getpid());

     /* Child will now execute the grep command. */

     execlp("grep", "grep", "test", "infile.txt", NULL);

     /* If the child process reaches this point, then  */
     /* execlp must have failed.                       */

     fprintf(stderr, "Child process could not do execlp.\n");
     exit(1);
  }
  else { /* Parent process. */
     if (child == (pid_t)(-1)) {
        fprintf(stderr, "Fork failed.\n"); exit(1);
     }
     else {
        c = wait(&cstatus); /* Wait for child to complete. */
        printf("Parent: Child  %ld exited with status = %d\n",
               (long) c, cstatus);
     }
  }
  return  0;
} /* End of main. */
```