Example1: The following sample program illustrates the operation of the fork() system call

#include <stdio.h>  
#include <unistd.h>  
  
int main()  
{  
    printf ("Before fork\n");  
    fork();  
    printf ("After fork\n");  
    return 0;  
}

Example 2: Use the **ps** utility with *appropriate options* (yes, you may have to read the man page again), observe and report the PIDs and the status of your processes. Provide a brief explanation.

#include <stdio.h>  
#include <stdlib.h>  
#include <unistd.h>  
  
int main (int argc, char \*argv[])  
{  
    int k, limit;  
    if (argc < 2) {  
        fputs ("Usage: must supply a limit value\n", stderr);  
        exit(1);  
    }  
    limit = atoi (argv[1]); /\* ascii to integer \*/  
    fork();  
    fork();  
    printf ("PID: %d\n", getpid());  
    for (k = 0; k < limit; k++)  
        printf ("%d\n", k);  
    return 0;  
}

Example 3 (Process termination): A process terminates its execution when the system call [exit()](http://www.cis.gvsu.edu/facilities/eos/man?Command=exit&Section=3) is invoked. To check the (termination) status of its child processes, a parent process has to call [wait()](http://www.cis.gvsu.edu/facilities/eos/man?Command=wait&Section=2) or [waitpid()](http://www.cis.gvsu.edu/facilities/eos/man?Section=2&Command=waitpid). The "status" parameter of these two system calls encodes the child exit status (the number passed to the exit() system call in the child process).

#include <unistd.h>  
#include <sys/wait.h>  
#include <stdio.h>  
#include <stdlib.h>  
  
int main () {  
    pid\_t pid, child;  
    int status;  
  
    pid = fork ();  
    if (pid < 0) {  
        perror ("fork failure");  
        exit (1);  
    }  
    else if (pid == 0) {        /\* code executed in child \*/  
        printf ("I am child PID %ld\n", (long) getpid ());  
        /\* insert an appropriate form of the exit() call here \*/  
    }  
    else {  
        /\* insert an appropriate form of wait() call here \*/  
        printf ("I am parent PID %d\n", getpid());  
        printf ("Child [PID %d] exit code is %d\n",  
                child, WEXITSTATUS (status));  
    }  
    return 0;  
}