EXERCISE 3

Process Status

process_status (“p3”)
p3 process is still running. PID= 5254

process_status (“p3”)
p3 process not found

Objectives:

- Write a function to determine if a given process is running.
- Learn to use Unix pipes and redirection with utl_process_spawn ( )

Process Status

Exercise Description:

The function, `process_status ( proc_name )`, checks the process list, determines if *proc_name* is in it, and indicates if the process is still running in the MSC/PATRAN command window. The function uses two temporary files called `xxps.com` and `xxps.out`. The function needs to check if either file exists, ask permission to use it, and deletes them when finished.

After inputing the file and executing the function a message in PATRAN should say:

```
p3 process is still running. PID= 5254
```

The PID (Process ID) number on your system will be different from 5254, but it should be an integer value.

Files:

All the files that used in this exercise are listed below. Each list includes the file, where it originated, and a summary of information of how it relates to the exercise.

File	Supplied/Created	Description
process_status.pcl	Created	The file should be able to write two temporary files and check if a process is still running.

Exercise Procedure:

1. Use the vi editor or jot to create a PCL function called, process_status(proc_name) in a file named `process_status.pcl`.

   There are quick reference pages for vi and UNIX in the appendix of this workbook for your reference.

2. Compile the function.

   Start the PCL compiler by typing

   `%p3pclcomp`

   in your xterm window.

3. Enter the command:

   `!!input process_status.pcl`

into the compiler's command line.

All the error messages and diagnostics will be written to the xterm. If the messages in the compiler say:

```
Compiling: process_status
Compiled: process_status
```

then there are no problems with your function.

4. Exit from the compiler.

   To exit the compiler type:

   exit

   or

   ctrl-d

5. Start MSC/PATRAN by typing **p3** at the xterm window.

6. Enter the following at the command line of PATRAN:

   !!input process_status.pcl

   If the name of your process is p3, then type:

   process_status ("p3")

7. You should get the following output:

```
p3 process is still running. PID= 5254
```

   The process id should be different for each machine.

Sample Solution:

```
FUNCTION process_status( proc_name )

/* Purpose: This function determines if a process is running.
 *
 * Input:
 * proc_name S Process name as issued at the command line.
 *
 * Output:
 * none
 *
 * Side Effects:
 * A file is created called xxps.out to temporarily record the
 * output of a ps command. If one exists, the user is asked for
 * overwrite permission.
 *
 * A message is written to history window
 * indicating the status of the process.
 * Note:
 * This is a machine dependent routine. This routine is written
 * for the “ps” command on the SGI machine. Check your machine
 * for differences.
 *
 */

    STRING proc_name[]

    INTEGER channel, pid, lrecl, length, status
    LOGICAL found
    STRING record[80], clength[5]


          length = str_length( proc_name )
          clength = str_from_integer( length )


          /*
          * Check for existence of files
          */

          IF( file_exists(“xxps.com”,””) ) THEN
              IF( ui_read_logical( “File xxps.com exists, Do you want to”// @
                         “overwrite it?” )) THEN
                  file_delete( “xxps.com”)
              ELSE
                  RETURN
              END IF
          END IF

          IF( file_exists(“xxps.out”,””) ) THEN
              IF( ui_read_logical( “File xxps.out exists, Do you want to”// @
                                 “overwrite it?” )) THEN
                  file_delete( “xxps.out”)
              ELSE
                  RETURN
              END IF
          END IF
```

```
        /*
        * Open a file and deposit process status command
        */

        text_open( “xxps.com”, “NRW”, 0, 0, channel )
        text_write_string( channel, “#! /bin/sh”)
        text_write_string( channel, “ps >xxps.out” )
        text_close( channel, “ “ )

        /*
        * Source the command file, then delete it
        */

        utl_process_spawn( “chmod +x xxps.com”, TRUE )
        utl_process_spawn( “./xxps.com”, TRUE )
        utl_process_spawn( “rm xxps.com”, TRUE )

        /*
        * Now open the file and parse to find process id
        */

        text_open( “xxps.out”, “OR”, 0, 0, channel )
        found = TRUE
        REPEAT loop
          status = text_read_string( channel, record, lrecl )

        /*
        * check for error read or end-of-file
        */

        IF( status != 0 ) THEN
            text_close( channel, “D” )
            found = FALSE
            BREAK loop
        END IF

        /*
        * The following line is specific to the SGI machine.
        * The column in which the process name is printed when
        * issuing the ps command varies but is column 4 on the SGI.
        * (it is column 5 on the SUN, for example )
        */

        UNTIL( str_token( record, “ “, 4, TRUE ) == proc_name )

        /*
        * if process found indicate the current status
        */

        IF( found ) THEN
            pid = str_to_integer( str_token( record, “ “, 1, TRUE ))
            ui_writef( “A”//clength//”,’ process is still running.”//@
                       “ PID =’,I8”, proc_name, pid )
            text_close(channel, “D”)
        ELSE
            ui_writef( “A”//clength//”,’ process not found’”, proc_name )
        END IF

    END FUNCTION
```