
Exercise 3

File Size

```
file_size ("settings.pcl")
```

The size of file settings.pcl is 1365 bytes.

```
file_size ("settings.pcl")
```

The file settings.pcl not found!

Objectives:

- Write a function to determine the size of a file in a directory
- Learn to use Unix pipes and redirection with `utl_process_spawn ()`



Exercise Description:

The function, `file_size (file_name)`, lists the files in the current working directory, finds the specific file, and writes out the name and file size to the history window. The function redirects the output of the `ls -l` command to a file called **xxps.out** which is then executed in the `xxps.com` file. The size of the file that you specify will be output to the history window of MSC.Patran. The function needs to check if either, `xxps.out` or `xxps.com`, exists, ask permission to use them, and then delete them when finished.

After inputing the file and executing the function a message in MSC.Patran should say:

```
The size of file settings.pcl is 1365 bytes.
```

The file can be any file that you have in the directory.

Files:

All the files that are 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
<code>file_size.pcl</code>	Created	The file should be able to write two temporary files and check the size of a file.

Exercise Procedure:

1. Open a text editor or jot to create a PCL function called, `file_size (file_name)` in a file named `file_size.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, or dos command window.

3. Enter the command:

```
!!input file_size.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: file_size
```

```
Compiled: file_size
```

then there are no problems with your function.

4. Exit from the compiler.

To exit the compiler type:

```
exit
```

or

```
ctrl-d
```

5. Find a file in your directory that you can check the file size.
6. Enter the following at the command line of MSC.Patran:

```
!!input file_size.pcl
```

If the name of your process is settings.pcl, then type:

```
file_size ( "settings.pcl")
```

7. You should get the following output:

```
The size of file settings.pcl is 1365 bytes
```

Sample Solution:

```

FUNCTION file_size( file_name )

/* Purpose: This function determines the size of the file in a directory.
*
* Input:
*     file_name  S  Process name as issued at the command line.
*
* Output:
*     none
*
* Side Effects:
*     A file is created called xxps.com to temporarily record the
*     output of a ls -l command. If one exists, the user is asked for
*     overwrite permission. For NT the command is dir /-C which will
*     eliminate the comma from the size of the file.
*
*     A message is written to history window
*     indicating the size of the file.
* Note:
*     Choose any file in the directory to find the size of the file.
*/

STRING      file_name[]
STRING      machine_type[32]

INTEGER     channel, size, lrecl, status
INTEGER     column = 9
INTEGER     size_column = 5
LOGICAL     found
STRING      record[80], size_str[20], result[20]

/*
* Check for existance of files
*/

sys_get_info(1,result)

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

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

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

```

```

text_open( "xxps.bat", "NRW", 0, 0, channel )

IF(machine_type == "winnt") THEN

text_write_string( channel, "dir /-C > xxps.out" )
text_close( channel, "" )
ELSE
text_write_string( channel, "#! /bin/sh" )
text_write_string( channel, "ls -l > xxps.out" )
text_close( channel, "" )
utl_process_spawn( "chmod +x xxps.bat", TRUE )
END IF
/*
 * Source the command file, then delete it
 */

utl_process_spawn( "xxps.bat", TRUE )
file_delete("xxps.bat")

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

text_open( "xxps.out", "OR", 0, 0, channel )
found = TRUE

IF(machine_type == "winnt") THEN
column = 4
size_column = 3
END IF

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

UNTIL( str_token( record, " ", column, TRUE ) == file_name )

/*
 * If the file is found it will return the size of the file.
 */

IF( found ) THEN
IF( result == "winnt") THEN
size = str_to_integer( str_token( record, " ", size_column, TRUE ) )
END IF
size_str = str_from_integer(size)
ui_write( "The size of file " //file_name// " is " //size_str// " bytes.")
ELSE
ui_write( "The file " //file_name// " was not found!")
END IF
text_close(channel, "D")
file_delete( "xxps.out" )

END FUNCTION

```


