PROJECT 1 - TEST CASES

(1) <u>Project Compilation and execution procedure</u> :

(1) The main program for the project is under “msh.c”
(2) Create a profile file titled “profile.src”.

A sample Profile file :

```
*************************
SIGN=$
PATH /usr/bin;/bin
HOME=/root
**************************
```

(3) Compile the source code in the Minix shell

In this we used Gcc compiler to execute our new sell. For this we must have gcc4.4.3 and gcc libraries to make our program run

We created a Makefile in this to automatically execute all the files to execute this we just type ‘make’ command to generate ‘.o’ files. This automatically generates object file using code written in Makefile

```
gcc msh.c -o msh.o
```

(4) Run the Project as “./msh” if compiled into a binary
or run “./msh”
(5) Run the commands as specified in the test cases

(2) TEST CASES :

(1)

TEST Case ID	M-001
Command Description	Shell Invoked from ash Shell
Procedure to execute	Compile the source code as described in section 1
Expected output	A new shell should be invoked with the msh> prompt
Project output	A new shell should invoked with the msh> prompt
Status	PASS

(2)

TEST Case ID	M-002
Command Description	Shell Invoked using the Profile file “profile.src”
Procedure to execute	Compile the source code as described in section 1 with a profile.src file present.
Expected output	1)If the file is not found or the information is not valid, the default values should be used. 2)If the file is found and the information is valid, the shell should load the environment vars.
Project output	Since the profile file was present, the new shell is invoked using the values set in the profile file
Status	PASS

(3)

TEST Case ID	M-003
Command Description	Shell Invoked without using the Profile file “profile.src”
Procedure to execute	Remove the profile file: “rm –rf profile.src” Compile the source code as described in section 1 without the profile.src file present
Expected output	1)If the file is not found or the information is not valid, the default values should be used. 2)If the file is found and the information is valid, the shell should load the environment vars.
Project output	Since the profile file was not present, we print out a warning message saying “profile.src : No such file “ The new shell is invoked using the default values set in the environment of the ash shell
Status	PASS

(4)

TEST Case ID	M-004
Command Description	Test for command “pwd”
Procedure to execute	After Log-in to the Project shell. Execute “pwd”
Expected output	When we log-in to the shell, we would be in the Home directory. The home directory specified in the “profile.src” file would be displayed.
Project output	Home directory i.e “/root” set in the profile file. Home directory displayed.
Status	PASS

(5)

TEST Case ID	M-005
Command Description	Test for command “ls”
Procedure to execute	Log-in to the Project shell. Execute “ls”
Expected output	The files and sub-directories in the current directory would be displayed.
Project output	The list of files and sub-directories are listed under the current directory.
Status	PASS

(6)

TEST Case ID	M-006
Command Description	Test for command “ls” with arguments
Procedure to execute	Log-in to the Project shell. Execute “ls –la”
Expected output	The files and sub-directories in the current directory would be displayed with complete details.
Project output	The list of files and sub-directories are listed under the current directory with complete details like file permissions, created time, Size of the files etc.
Status	PASS

(7)

TEST Case ID	M-007
Command Description	Test for command “cat”
Procedure to execute	Log-in to the Project shell. (1)Execute “cat > test” (2) Enter some text at the prompt (3) Press “Ctrl d” (4) Execute “cat test”
Expected output	An editor is opened to enter values Once “Ctrl d” is pressed the file is created with the contents typed.
Project output	File ‘test’ created with the contents typed present in the file. Upon execution of step (4), the contents of the file is displayed
Status	PASS

(8)

TEST Case ID	M-008
Command Description	Test for command “cat” for overwriting
Procedure to execute	Log-in to the Project shell. (1)Execute “cat > test”. Assume “test” file exists (2) Enter some text at the prompt (3) Press “Ctrl d” (4) Execute “cat test”
Expected output	An editor is opened to enter values Once “Ctrl d” is pressed the file is overwritten with the contents typed.
Project output	File “test” saved with the contents typed present in the file. Upon execution of step (4), the over-written contents of the file is displayed. Old content in the file are not present.
Status	PASS

(9)

TEST Case ID	M-009
Command Description	Test for command “ps”
Procedure to execute	Log-in to the Project shell. (1)Execute “ps” (2) execute “ps –ef”
Expected output	The list of processes is displayed
Project output	The Process is listed in a brief manner with first execution. With (2) step, the complete details of all the processes are displayed.
Status	PASS

(10)

TEST Case ID	M-010
Command Description	Test for Ctrl+C handler
Procedure to execute	Log-in to the Project shell. (1)Press ctrl+c
Expected output	The Shell prompts whether to terminate or not. If pressed “yes”, the shell terminates gracefully. If pressed “no” , the shell is returned.
Project output	The Prompt asking to confirm exit is displayed. Upon typing “no”, the shell prompt is returned. Upon typing “yes”, the shell exits and goes to “ash” shell prompt.
Status	PASS

(11)

TEST Case ID	M-011
Command Description	Negative Test for an invalid command
Procedure to execute	Log-in to the Project shell. (1)execute “abc”. Make sure no file with name “abc” exists
Expected output	An error message stating “abc” not found is displayed.
Project output	An error message stating “abc” not found is displayed. And the shell returns to the prompt for further executions.
Status	PASS

(12)

TEST Case ID	M-012
Command Description	Negative Test for an invalid “ls” command
Procedure to execute	Log-in to the Project shell. (1)execute “ ls xyz”. Make sure no file with name “xyz” exists
Expected output	An error message is displayed specifying absence of any such file “xyz”
Project output	An error message stating “xyz : No such file or directory” is displayed.
Status	PASS

(13)

TEST Case ID	M-013
Command Description	Negative Test for an invalid “cat” command
Procedure to execute	Log-in to the Project shell. (1)execute “ cat xyz”. Make sure no file with name “abcde” exists
Expected output	An error message is displayed specifying absence of any such file “xyz”
Project output	An error message stating “xyz : No such file or directory” is displayed.
Status	PASS

(14)

TEST Case ID	M-014
Command Description	Test for Piped Command
Procedure to execute	Log-in to the Project shell. (1)execute “ cat abc”. (2)Enter “Minix Project” in the editor and save the file. (3) Now execute “cat abc | wc”
Expected output	The number of lines, number of words and number of characters in the file “abc” are displayed.

Project output	The Pipe command worked properly. The output displayed as,
	`Mysh> cat abc | wc`
	`1    2 14`
	Where “1” – number of lines
	“2” – Number of words
	“14” – No of characters

Status	PASS

(15)

TEST Case ID	M-015
Command Description	Test for “spawn process”
Procedure to execute	Log-in to the Project shell. (1)”ps” (2)Now execute “fork_test”
Expected output	creates a child process and echo the process id of the new process and then exits
Project output	(1) “ps” execution showed “ps” commands pid as “1572” (2) Upon execution of “fork_test”, the output displayed : Msh> fork_test Process ID : 1577 Thus a new child process created and its PID displayed. The shell returns to its prompt meaning it exited from the process
Status	PASS

(16)

TEST Case ID	M-016
Command Description	Test for evaluating result
Procedure to execute	Res=(1+2*3/2)
Expected output	This evaluates the expression and prints the desired output
Project output	The result is 4
Status	PASS

(17)

TEST Case ID	M-015
Command Description	Test for $i

Procedure to execute	put=$i=4
	put=$i=10

Expected output	Stores the value of $i in memory and retrieve the latest value of $i. It displays the value even if we exit shell and run back later.
Project output	To retrieve the value stored, we need to use the following command get=$i then the result is displayed as The value of I is 10
Status	PASS