What is Shells?

**A Shell** provides you with an interface to the Unix system. It gathers input from you and executes programs based on that input. When a program finishes executing, it displays that program's output.

Shell Types

In Unix, there are two major types of shells −

* **Bourne shell** − If you are using a Bourne-type shell, the **$** character is the default prompt.
* **C shell** − If you are using a C-type shell, the % character is the default prompt.

The Bourne Shell has the following subcategories −

* Bourne shell (sh)
* Korn shell (ksh)
* Bourne Again shell (bash)
* POSIX shell (sh)

The different C-type shells follow −

* C shell (csh)
* TENEX/TOPS C shell (tcsh)

Bourne shell is usually installed as **/bin/sh** on most versions of Unix. For this reason, it is the shell of choice for writing scripts that can be used on different versions of Unix.

Shell Scripts

The basic concept of a shell script is a list of commands, which are listed in the order of execution. A good shell script will have comments, preceded by **#** sign, describing the steps.

Example Script

Assume we create a **test.sh** script. Note all the scripts would have the **.sh** extension. Before you add anything else to your script, you need to alert the system that a shell script is being started. This is done using the **shebang** construct. For example −

* #!/bin/sh
* This tells the system that the commands that follow are to be executed by the Bourne shell. *It's called a shebang because the* ***#*** *symbol is called a hash, and the ! symbol is called a bang*.
* To create a script containing these commands, you put the shebang line first and then add the commands −
* #!/bin/bash
* pwd
* ll
* Shell Comments
* You can put your comments in your script as follows −
* #!/bin/bash
* # Author : Vimlesh
* # Script follows here:
* pwd
* ll
* Save the above content and make the script executable −
* $chmod +x test.sh
* The shell script is now ready to be executed −
* $./test.sh

Upon execution, you will receive the current path and list of all files and folder in current directory

**Note** − To execute a program available in the current directory, use **./program\_name**

Extended Shell Scripts

The shell is, after all, a real programming language, complete with variables, control structures, and so forth. No matter how complicated a script gets, it is still just a list of commands executed sequentially.

The following script uses the **read** command which takes the input from the keyboard and assigns it as the value of the variable USR and finally prints it on STDOUT.

* #!/bin/sh
* echo "What is your name?"
* read USR
* echo "Hello, $URS"

read VARNAME #VARNAME IS TEMP VARIABLE WHERE DATA CAN BE STORED, read is function to read data from console/terminal

echo $VARNAME #ACCESS THE VARIABLE

* Here is a sample run of the script −
* $./test.sh
* What is your name?
* Vimlesh
* Hello, Vimlesh
* $