1. How to determine Shell

You can get the name of your shell prompt, with following command :

**Syntax:**

echo $SHELL

# Shell Scripting She-bang

The sign #**!** is called she-bang and is written at top of the script. It passes instruction to program **/bin/sh.**

**Example:**

1. #!/bin/bash
2. echo Hello World
3. #!/bin/ksh
4. echo Hello World

# Shell Scripting Comments

Any line starting with a hash (#) becomes comment. Comment means, that line will not take part in script execution. It will not show up in the output.

# Shell Scripting Variables

Scripts can contain variables inside the script.



Look at the above snapshot, two variables are assigned to the script **$var1** and **$var2.**

As scripts run in their own shell, hence variables do not survive the end of the script.



Look at the above snapshot, **var1** and **var2** do not run outside the script.

# Shell Scripting Sourcing a file

A file is sourced in two ways. One is either writting as **source** <**fileName**> or other is writting as ./<**filename>** in the command line. When a file is sourced, the code lines are executed as if they were printed on the command line.

# Troubleshooting a shell script

There is one more way other than script execution to run a script in a different shell. Type bash with the name of the script as parameter.

**Syntax:**

1. bash **<fileName>**

# Shell Script Parameters

A bash shell script have parameters. These parameters start from **$1** to **$9.**

When we pass arguments into the command line interface, a positional parameter is assigned to these arguments through the shell.

The first argument is assigned as $1, second argument is assigned as $2 and so on...