SHRI G.S. INSTITUTE OF TECHNOLOGY & SCIENCE
Department of Computer Engineering
C02_____: Operating system
Lab Assignment–II

Bash and Bash scripts

What is Bash?

Bash is the shell, or command language interpreter, for the GNU operating system. The name is an acronym for the '`Bourne-Again SHell`', a pun on Stephen Bourne, the author of the direct ancestor of the current Unix shell `sh`, which appeared in the Seventh Edition Bell Labs Research version of Unix.

What is a shell?

At its base, a shell is simply a macro processor that executes commands. The term macro processor means functionality where text and symbols are expanded to create larger expressions.

A Unix shell is both a command interpreter and a programming language. As a command interpreter, the shell provides the user interface to the rich set of GNU utilities. The programming language features allow these utilities to be combined. Files containing commands can be created, and become commands themselves. These new commands have the same status as system commands in directories such as `/bin`, allowing users or groups to establish custom environments to automate their common tasks.

Shells may be used interactively or non-interactively. In interactive mode, they accept input typed from the keyboard. When executing non-interactively, shells execute commands read from a file.

A shell allows execution of GNU commands, both synchronously and asynchronously. The shell waits for synchronous commands to complete before accepting more input; asynchronous commands continue to execute in parallel with the shell while it reads and executes additional commands. The *redirection* constructs permit fine-grained control of the input and output of those commands. Moreover, the shell allows control over the contents of commands' environments.

Shells also provide a small set of built-in commands (*builtins*) implementing functionality impossible or inconvenient to obtain via separate utilities. For example, `cd`, `break`, `continue`, and `exec` cannot be implemented outside of the shell because they directly manipulate the shell itself. The `history`, `getopts`, `kill`, or `pwd` builtins, among others, could be implemented in separate utilities, but they are more convenient to use as builtin commands. All of the shell builtins are described in subsequent sections.

While executing commands is essential, most of the power (and complexity) of shells is due to their embedded programming languages. Like any high-level language, the shell provides variables, flow control constructs, quoting, and functions.

Goal:To know about the Shell and Bash

- Where is the bash program located on your system?
- Use the --version option to find out which version you are running
- Which shell configuration files are read when you login to your system using the graphical user interface and then opening a terminal window?
- Are the following shells interactive shells? Are they login shells?
  • A shell opened by clicking on the background of your graphical desktop, selecting “Terminal” orsuch from a menu
  .• A shell that you get after issuing the command ssh localhost.
  • A shell that you get when logging in to the console in text mode.
  • A shell obtained by the command xterm &.
  • A shell opened by the mysystem.sh script.
  • A shell that you get on a remote host, for which you didn't have to give the login and/or passwordbecause you use SSH and maybe SSH keys.
- Can you explain why bash does not exit when you type Ctrl+C on the command line?
  6.Display directory stack content
- If it is not yet the case, set your prompt so that it displays your location in the file system hierarchy,for instance add this line to ~/.bashrc:export PS1="\u@\h \w> "
- Display hashed commands for your current shell session.
- How many processes are currently running on your system? Use ps and wc, the first line of output ofps is not a process
- How to display the system hostname?Only the name.