Birla Institute of Technology & Science, Pilani
Pilani Campus

II SEMESTER 2020-2021
Assignment-1

Course No.: **IS F462** **Course Title:** **Network Programming**
Deadline: **As on Canvas** **Maximum Marks:** **60M (15%)**

Note:

- Maximum of three students per group. Upload code in Canvas.
- Name your file idno1_idno2_idno3_assignment1.tar .

P1. You are required to build a bash-like shell for the following requirements. Your program should not use temporary files, popen(), system() library calls. It should only use system-call wrappers from the library. It should not use *sh* or *bash* shells to execute a command.

a) Shell should wait for the user to enter a command. User can enter a command with multiple arguments. Program should parse these arguments and pass them to execv() call. For every command, shell should search for the file in PATH and print any error. Shell should also print the pid, status of the process before asking for another command.

b) shell should create a new process group for every command. When a command is run with & at end, it is counted as background process group. Otherwise it should be run as fore-ground process group (look at tcsetpgrp()). That means any signal generated in the terminal should go only to the command running, not to the shell process.

c) shell should support <, >, and >> redirection operators. Print details such as fd of the file, remapped fd.

d) shell should support any number of commands in the pipeline. e.g. ls|wc|wc|wc. Print details such as pipe fds, process pids and the steps. Redirection operators can be used in combination with pipes.

e) shell should support two new pipeline operators "||" and "|||". E.g.: ls -l || grep ^-, grep ^d. It means that output of ls -l command is passed as input to two other commands. Similarly, "|||" means, output of one command is passed as input to three other commands separated by ",".

f) shell should support a mode called 'short-cut commands' executed by command sc. In this mode, a command can be executed by pressing Ctrl-C and pressing a number. This number corresponds to the index in the look up table created and deleted by the commands `sc -i <index> <cmd>/ sc -d <index> <cmd>`.

Deliverables:

- Brief Design Document (.pdf)
- shell.c

[20 M]

P2. Cluster Shell. In this problem you are required to extend the shell features to a cluster of machines, each identified by a name. The name to ip mapping is available in a config file, whose path is specified at the start of the shell. Assume that N nodes in the cluster are named as n1, n2 ..... nN.

- Cluster shell is run on any one of the nodes. When a command is run e.g. `ls` it executed on the local system. When n2.ls is run in n1, it is executed on node n2 and the output is listed on n1. When `n*.ls` is run on n1, `ls` is run on all nodes and output is displayed on n1. This applies to other commands as well. By default, all commands on a remote node are executed in the home directory of the user logged in n1. That is, it is necessary to have the same user on all systems. When `n2.cd <path>` or `n*.cd <path>` is executed, directory is changed.
- When the command `n1.cat file|n2.sort|n3.uniq` is executed on n5, the commands are executed on different nodes taking input from the previous command but the last output is displayed on the node n5 it is executed on.
- The command `nodes` display the list of nodes (name, ip) currently active in the cluster.

Deliverables:

- clustershell_client.c, clustershell_server.c
- pdf file explaining design decisions

[20M]

P3. Message Queues: Using message queues design and implement a group messaging system. It should be able to allow users in a UNIX system to create groups, list groups, join, send private messages, send message to groups, receive message from groups in online/offline modes. A user may set auto delete <t> option which means, users who joined after t seconds from the time of message creation, to them message will not be delivered.

Deliverables:

- msgq_server.c, msgq_client.c
- pdf file explaining design decisions

[20M]

--&--