PRCMP P07

The Unix shell: working with the shell

May 2023

1 Viewing file contents

Some useful commands to determine the type of files and view their contents. Read the respective pages of the manual for details.

file – Determines the type of file by analysing its content.

more – Shows the content of a file, page by page.

less – Shows the content of a file, page by page. It is more efficient than the **more** command with large files.

cat – Concatenates (joins) and prints files. The contents of the file are dumped to the output.

head – Displays the first lines of a file.

tail – Displays the last lines of a file.

Questions

1. With the `nano` editor, create the `dummy.txt` file and add a few lines of text. After saving the file, type and run the `file dummy.txt` command line. Interpret the result.
2. Rename the file `dummy.txt` to `dummy.pdf`. Repeat the `file` command for the file with the new name. Interpret the result.
3. Create the new file `long.txt` using the program `nano` and fill it with about 40 lines of text. Use the `cat` command to see the contents of the `long.txt` file. Would this way of seeing the file contents be appropriate if the file was much longer?
4. Use the `more` command to see the contents of `long.txt` and then do the same with the `less` command. How do these commands differ from `cat`?
5. Read the `head` command manual page. Use this command with the `long.txt` file. Is the result as expected?
6. Repeat the previous question, but print only the first three lines of the file.
7. Read the `tail` command manual page. Use this command with the `long.txt` file. Again, is the result as expected?
8. Repeat the previous question, but print only the last seven lines of the file.
9. The `cat` command is used to concatenate (i.e. join in sequence) text from various sources. Create the file `dumdum.txt` and add a few lines of text. Use the `cat` command to concatenate the contents of the `dummy.txt` and `dumdum.txt` files in this order. What do you observe from the command line result?
10. Now, concatenate the `dumdum.txt` and `dummy.txt` files, in this order. Is there any difference in the result from the previous question?

2 Redirecting IO

Questions

11. Redirect the result of the `man ls` command to a the new `man_ls.txt` file. View the contents of this file.

    man ls > man_ls.txt

12. Read the manual page for the `echo` command.

    (a) Run the following commands sequentially:

    ```
    echo 'one' > file.tmp
    echo 'two' > file.tmp
    ```

    What is the content of `file.tmp`?

    (b) Now, repeat the commands, with the variation indicated in the second line:

    ```
    echo 'one' > file.tmp
    echo 'two' >> file.tmp
    ```

    What is the content of `file.tmp`? Explain the difference to the result of the previous question.

13. Read the manual page for the `sort` command.

    (a) Using a text editor, create the `words.txt` file with several lines of text.

    (b) Run the following command:

    ```
    sort < words.txt > sorted_words.txt
    ```

    Analyse this command line. Is the result in accordance with your analysis?

    (c) Make a duplicate of the `words.txt` file, named `duplicate.txt` and check its content.

    Now, run the following command line:

    ```
    sort < duplicate.txt > duplicate.txt
    ```

    Check again the contents of the `duplicate.txt` file. Explain the result.

14. `/dev/null` is a virtual output device where we can send output that is to be discarded (i.e. not displayed or saved).

    (a) Try this virtual device, with the following command line:

    ```
    ls /bin > /dev/null
    ```

    Find an explanation for what happened.

    (b) Run both command lines:

    ```
    ping
    ping -c 3 www.google.com
    ```

    Do both command lines produce output?

    (c) Repeat both command lines, but this time redirect the stderr (exclusively) to /dev/null.

    Was there any change in the output of these commands? Find a justification for what you observed.

3 Pipes

Unix allows you to create command pipelines using pipes: e.g. `cmd1 | cmd2 | cmd3` A pipe connects the stdout of a command to the stdin of the next command on the command line.

Note that both programs run at the same time.

Questions

15. Alphabetically sort the lines of the manual page of the `ping` command and place the result in the `manual_ls_sorted.txt` file.
16. Use the `wc` command to count the lines, words and characters on the `chmod` command manual page.
17. The `tr` utility copies the standard input to the standard output with substitution or deletion of selected characters. Read the respective page of the manual.
    (a) Use a text editor to create a file with several lines of text.
    (b) Using the tr command, replace all the lowercase letters of the file you created with uppercase, checking the result on the monitor.

        If the result is correct, repeat the command, but save the result in the file `result1.txt`.
    (c) Delete all vowels from the `result1.txt` file.

        After checking on the monitor that the command line is correct, save the result in the `result2.txt` file.
18. Create a command pipeline that, in the end, meets all of the following requirements:
    - list the contents of the /bin directory in long format,
    - squeezes consecutive repeated spaces (hint: use the `tr` command),
    - selects the columns featuring (1) the permissions, (2) the name of the owner and (3) the file name (hint: use the `cut` command),
    - shows only the last 10 lines of the list.

4 Environment variables

Some useful commands to visualise environment variables. Read the respective pages of the manual for details.

env – Print global environment variables, or set environment and execute command.

printenv – Print global environment variables, or a specific variable.

set – Print all environment variables.

unset – Delete a local variable.

Questions

19. List the global environment values set in your session.
    (a) What is the value of `HOME`?
    (b) What is the current value of `PWD`?
    (c) Change to `/usr/bin` directory. Print the current value of `PWD`. What could have caused it to change?
    (d) What is the current value of `HOME`? Why hasn't it changed, as `PWD` did?
    (e) List the files in your current directory. Now list the files in your home directory, using the `HOME` variable as a parameter.
    (f) Change back to your home directory.
20. List all your environment variables (global and local).

(a) Local variables `LINES` and `COLUMNS` tell you the size of your terminal window in the number of characters. Write down their values, and then resize your terminal window. Read, again, their values issuing the command:
`echo "Screen size: $COLUMNS x $LINES"`

(b) Variable `RANDOM` provides a new number in the range [0, 32767]. Check that the behavior of the variable is as expected by launching the command line `echo $RANDOM` several times.

How do you explain what you observed?

(c) Assigning a value to this variable seeds the pseudo-random number generator: *i.e.* the following random numbers are generated from this seed.

Assign an arbitrary number to the `RANDOM` variable, using the `=` operator. Then, generate a sequence of 3 random numbers.

Again, assign the same value to the `RANDOM` variable and re-generate a sequence of 3 random numbers.

How do you explain what you observed?

21. The `ping` utility is a program that sends ICMP echo requests to a remote machine to test its reachability. *Note that not all machines respond to echo requests (by configuration) to hide their presence or to avoid Denial-of-Service attacks.*

(a) Create two new variables – `site1` and `site2` – assigning them values `"www.google.com"` and `"www.not.a.website.com"`, respectively.

(b) Ping the Google website with only one echo request using the variable you just set:
`ping -c 1 \$site1`
Did the website replied?

(c) Do the same action for the second website. Did the website replied?

(d) Repeat the previous two commands, but now write a success message if the remote machine responds or a failure message otherwise.
Hint: use the `&&` and `||` operators.

(e) Delete variables `site1` and `site2`. Check that both variables are no longer part of the environment.

5 Job control

Questions

22. Read the manual page for command `sleep`.

(a) Launch the `sleep` command to suspend execution for 1000 seconds. The command should run in the foreground.

(b) Suspend the execution of the foreground job by pressing the appropriate keys so that you can gain access to the shell again.

(c) List the active jobs that belong to your shell session. What is job number and the current status of the `sleep` job?

(d) Resume execution of this job in the background. Once more, list the active jobs in your session. Did the status of your job change?

(e) Bring back the job into the foreground, using the `fg` command. Interrupt the execution of your job by pressing the appropriate keys. List the current jobs in your session and check the status of the `sleep` job. Give a reason for what you observed.

23. The `curl` program is a tool to transfer data from or to a server that supports many usual protocols, including HTTP.

(a) Download "The Complete Works of William Shakespeare" from the Project Gutenberg website, issuing the following command:

```
curl -s -XGET https://www.gutenberg.org/cache/epub/100/pg100.txt
```

(b) Repeat the same action, but run the command in the background.

Were you able to effectively use the shell while transferring the document? Why?

(c) Finally, download the document and save it directly to the `WorksOfShakespeare.txt` file.

What do you think about running an interactive job in the background?

24. Launch three `sleep` commands in the background to sleep for 500 seconds, each.

(a) Use the `ps` command to view the processes and write down their process identifiers (PIDs).

(b) Use the `kill` command with the PIDs to terminate each of the three processes. Check their status by re-running the ps command.