**CSc 3320: Systems Programming**

Spring 2021

Homework

# 3: Total points 100

Submission instructions:

1. Create a Google doc for each homework assignment submission.
2. Start your responses from page 2 of the document and copy these instructions on page 1.
3. Fill in your name, campus ID and panther # in the fields provided. If this information is missing in your document TWO POINTS WILL BE DEDUCTED per submission.
4. Keep this page 1 intact on all your submissions. If this *submissions instructions* page is missing in your submission TWO POINTS WILL BE DEDUCTED per submission.
5. Each homework will typically have 2-3 PARTS, where each PART focuses on specific topic(s).
6. Start your responses to each PART on a new page.
7. If you are being asked to write code copy the code into a separate txt file and submit that as well.
8. If you are being asked to test code or run specific commands or scripts, provide the evidence of your outputs through a screenshot and copy the same into the document.
9. Upon completion, download a .PDF version of the document and submit the same.

Full Name: Hakan Gunerli

Campus ID: hgunerli1

Panther #: 002504797

**10 pts for the neatness factor of your presentation.**

**PART 1: 30pts**

1. For each command tryout at least one example provided in **Chapter 3** of the Unix textbook. Feel free to use your own example. Show the screenshot for each command’s output. Present your output in a tabular form with column 1 as index (1,2,3..), second column as the command, third as the usage, fourth as the screenshot of the output. You can just show a small snapshot for the output -- we do not need the entire screen’s image.

**Part II : 30pts**

1. For each command tryout at least one example provided in **Chapter 4** of the Unix textbook. Feel free to use your own example. Show the screenshot for each command’s output. Present your output in a tabular form with column 1 as index (1,2,3..), second column as the command, third as the usage, fourth as the screenshot of the output. You can just show a small snapshot for the output -- we do not need the entire screen’s image.

**Part III : 30pts**

1. For each command tryout at least one example provided in **Chapter 5** of the Unix textbook. Feel free to use your own example. Show the screenshot for each command’s output. Present your output in a tabular form with column 1 as index (1,2,3..), second column as the command, third as the usage, fourth as the screenshot of the output. You can just show a small snapshot for the output -- we do not need the entire screen’s image.

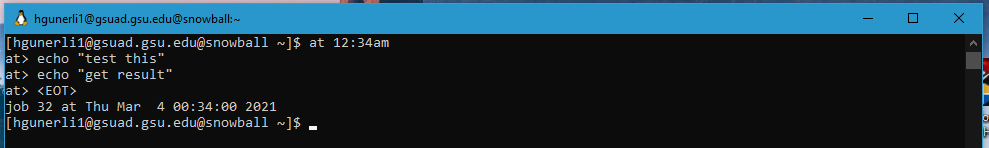
**Part I: Chapter 3**

1. **at**

Command name: at

Usage: allows you to schedule or delete jobs based on time.

Output:

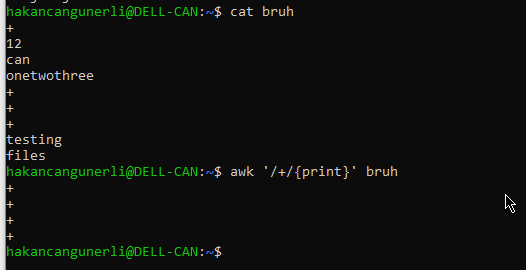


1. **awk**

**Command name: awk**

**Usage: finding patterns in files/texts.**

**Output:**

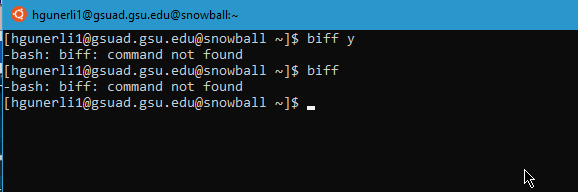
****

1. **biff**

**Command name: biff**

**Usage: mail notification, does not exist**

**Output:**

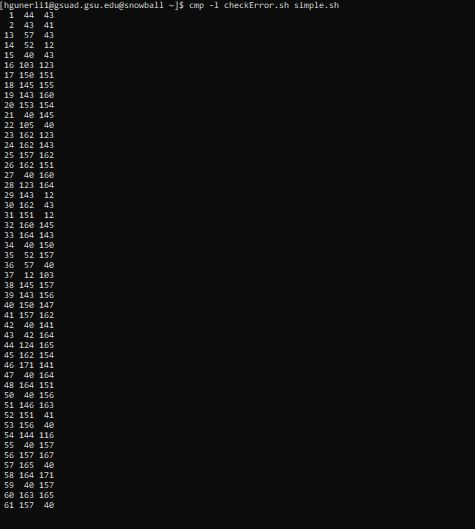
****

1. **cmp**

**Command name: cmp**

**Usage: check two files for equality byte by byte.**

**Output:**

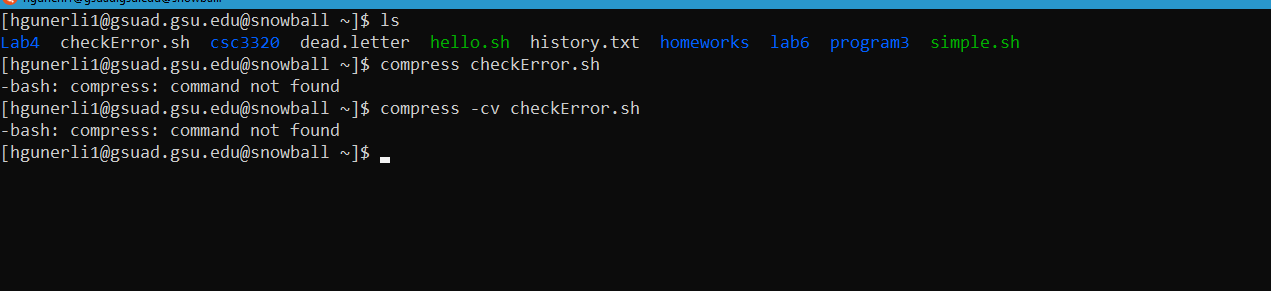
****

1. **compress**

**Command name: compress**

**Usage: compacts the file.**

**Output:**

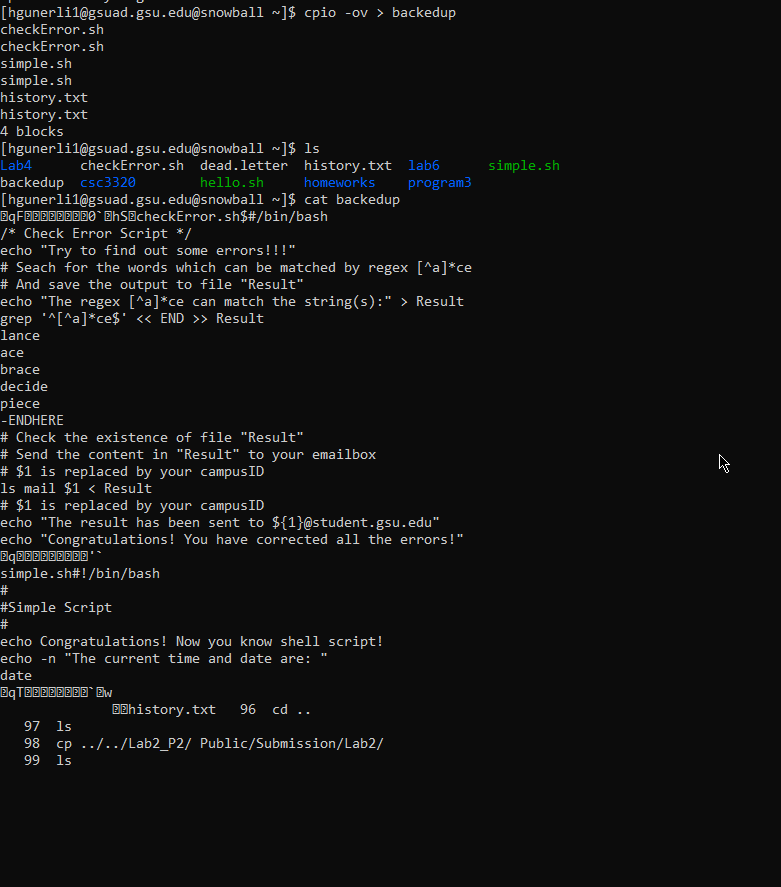
****

1. **cpio**

**Command name: cpio**

**Usage: create a backup files**

**Output:**

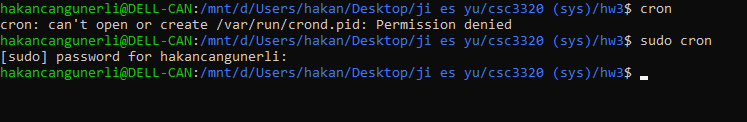
****

1. **cron**

**Command name: cron**

**Usage: scheduled jobs, similar to at**

**Output:**

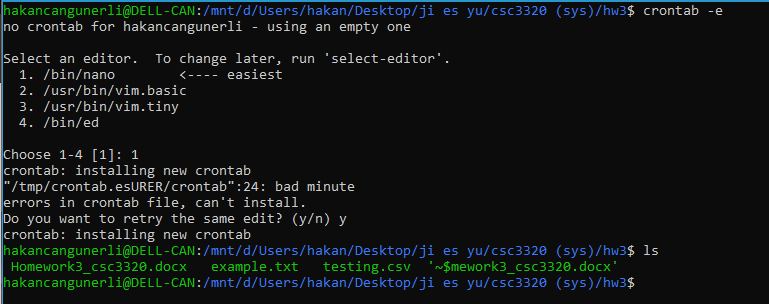
****

1. **crontab**

**Command name: crontab**

**Usage: schedule jobs at a scheduled time**

**Output:**

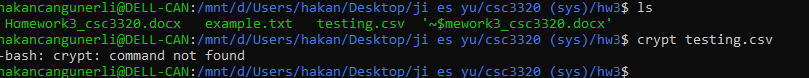
****

1. **crypt**

**Command name: crypt**

**Usage: encrypt files, however it is not used that much anymore since it is easy to break for today’s standards .**

**Output:**

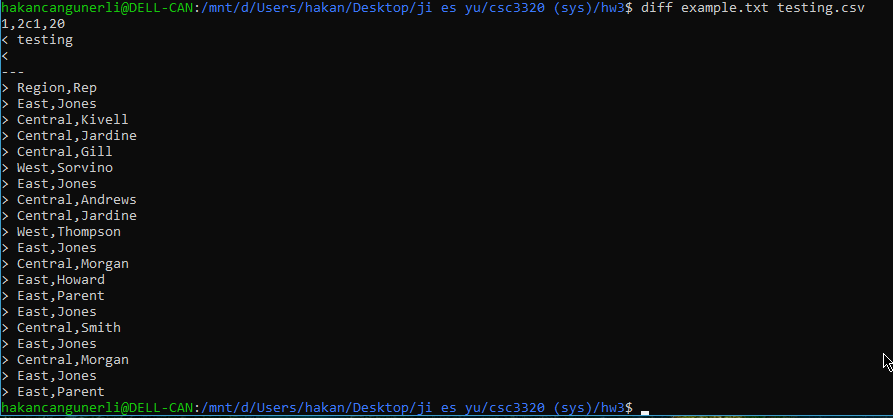
****

1. **diff**

**Command name: diff**

**Usage: compare and find differences between files**

**Output:**

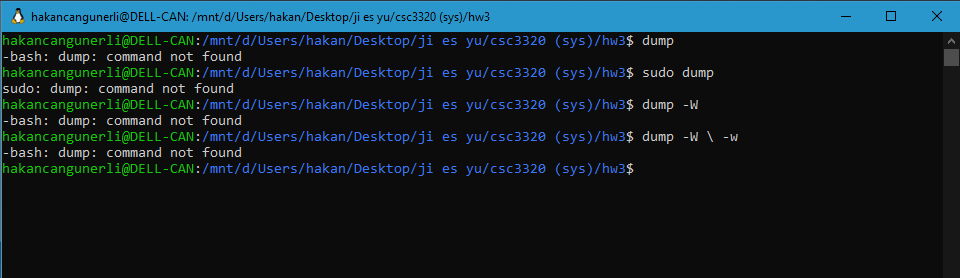
****

1. **dump**

**Command name: dump**

**Usage: backed up file systems**

**Output:**

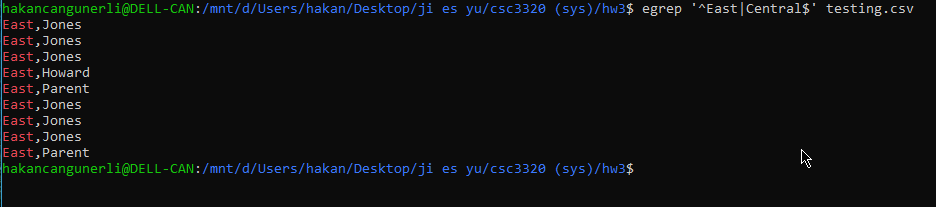
****

1. **egrep**

**Command name: egrep**

**Usage: similar to grep and fgrep, match pattern this one has extended regex.**

**Output:**

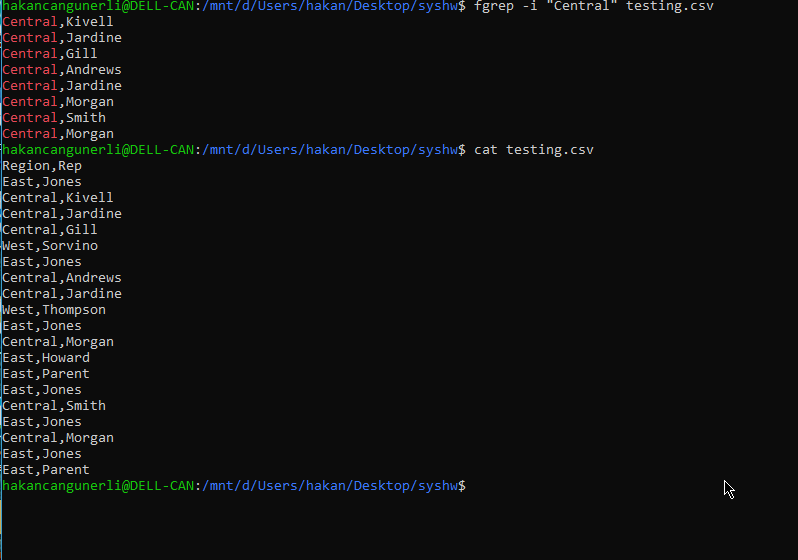
****

1. **fgrep**

**Command name: fgrep**

**Usage: filter fixed character strings**

**Output:**

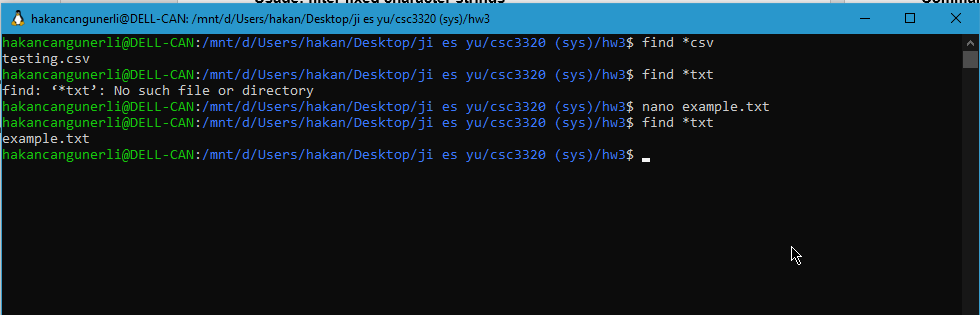
****

1. **find**

**Command name: find**

**Usage: search for files**

**Output:**

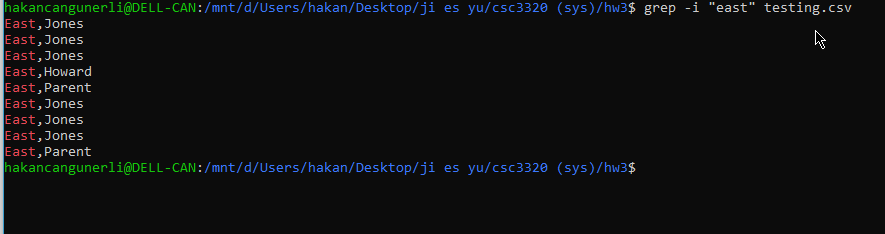
****

1. **grep**

**Command name: grep**

**Usage: print lines that match patterns, like fgrep except fgrep is not for regexes.**

**Output:**

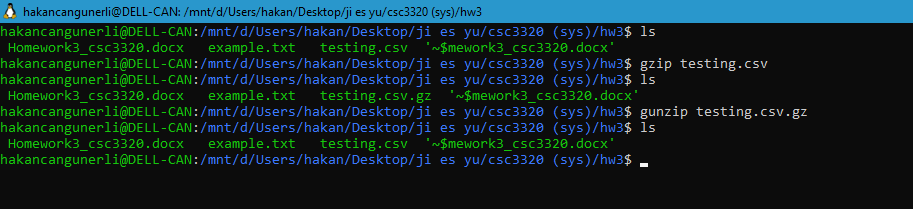
****

1. **gunzip**

**Command name: gunzip**

**Usage: decompress files, antagonist to gzip**

**Output:**

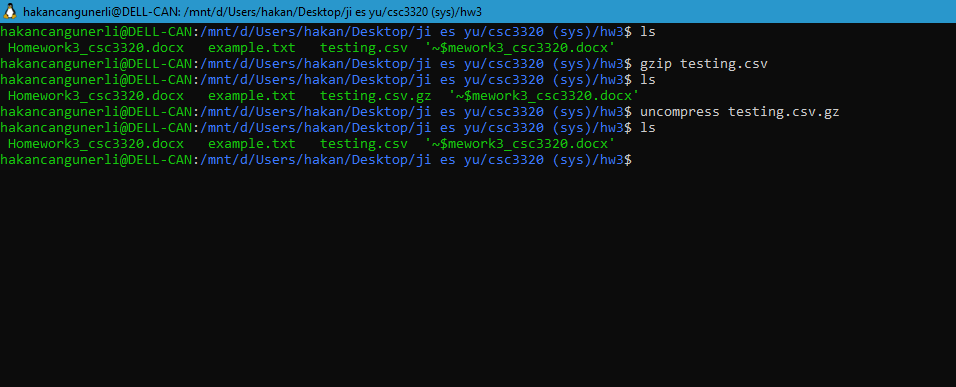
****

1. **gzip**

**Command name: gzip**

**Usage: compress files, similar to tar.**

**Output:**

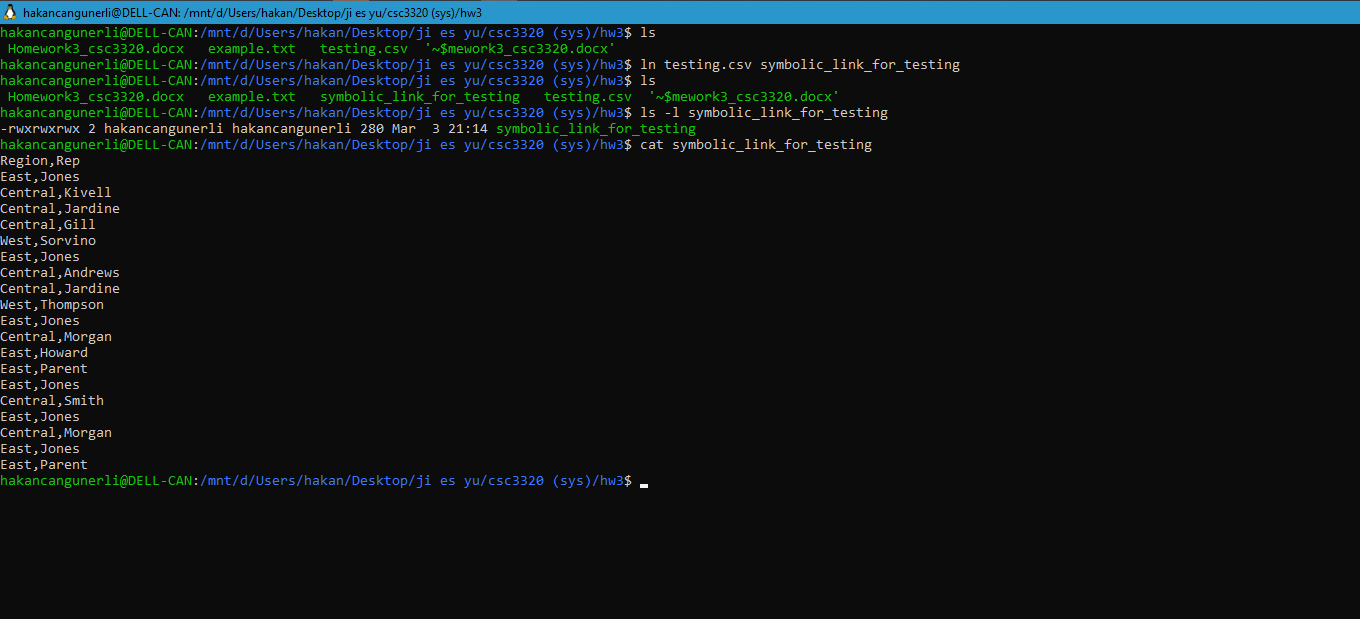
****

1. **ln**

**Command name: ln**

**Usage: linking files.**

**Output:**

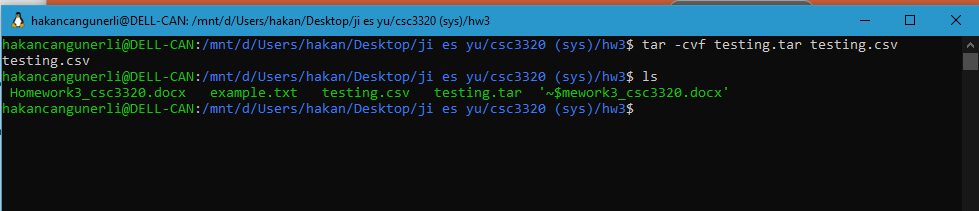
****

1. **tar**

**Command name: tar**

**Usage: for compression**

**Output:**

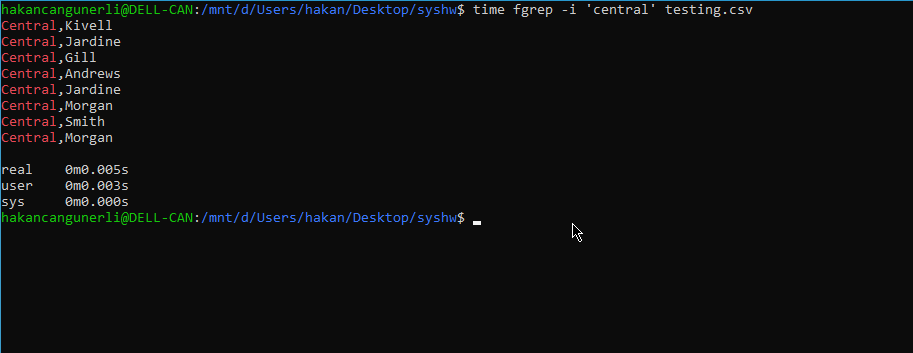
****

1. **time**

**Command name: time**

**Usage: how long a command takes to run.**

**Output:**

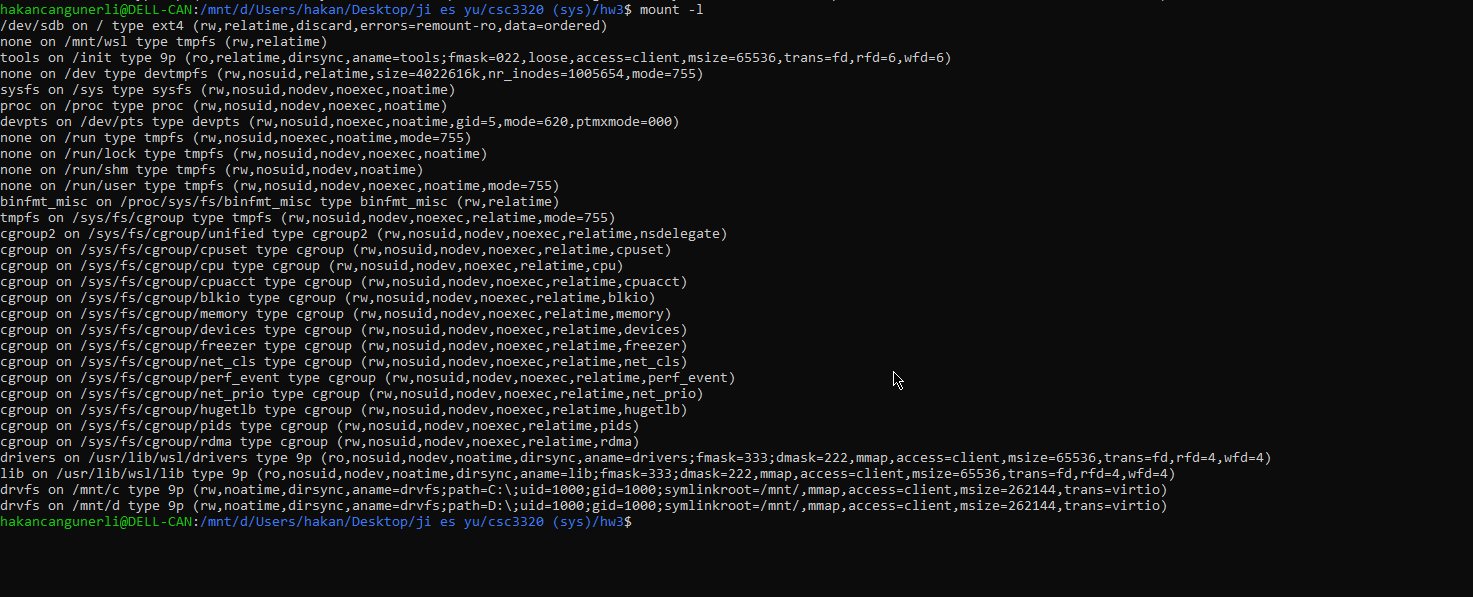
****

1. **mount**

**Command name: mount**

**Usage: mount filesystems.**

**Output:**

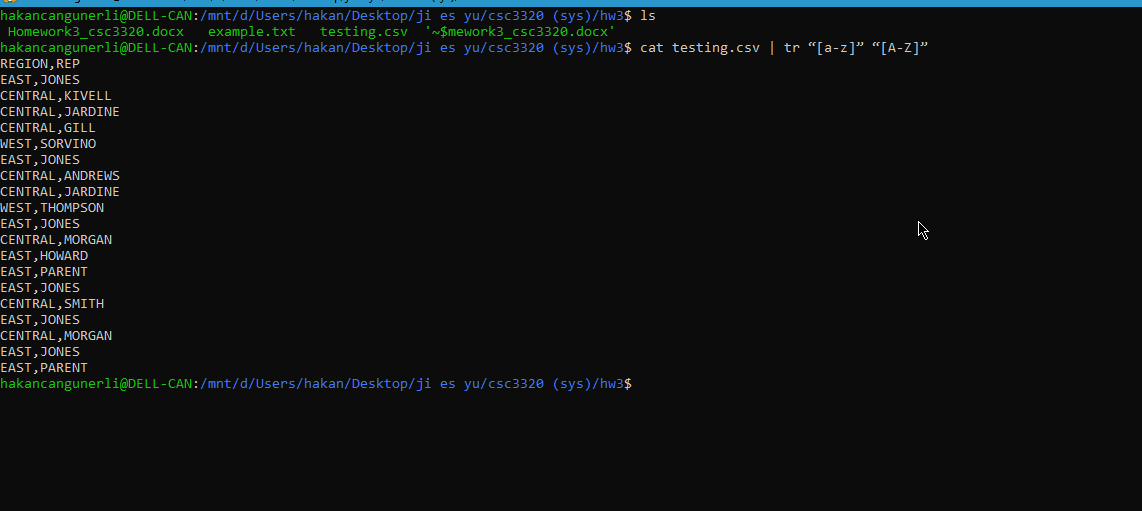
****

1. **tr**

**Command name: tr**

**Usage: translate and delete characters**

**Output:**

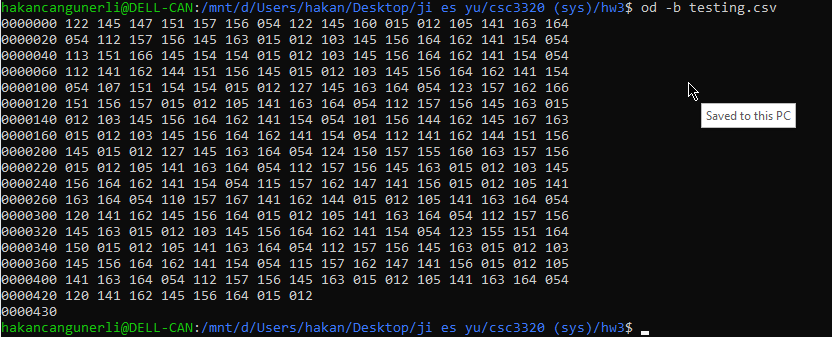
****

1. **od**

**Command name: od**

**Usage: output content as octal values**

**Output:**

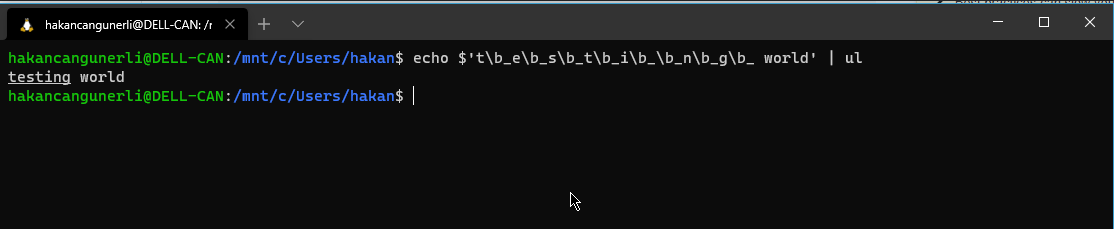
****

1. **ul**

**Command name: ul**

**Usage: underline text**

**Output:**

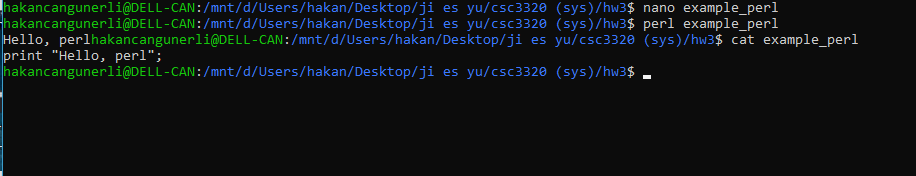
****

1. **perl**

**Command name: perl**

**Usage: writing commands using perl, a programming language.**

**Output:**

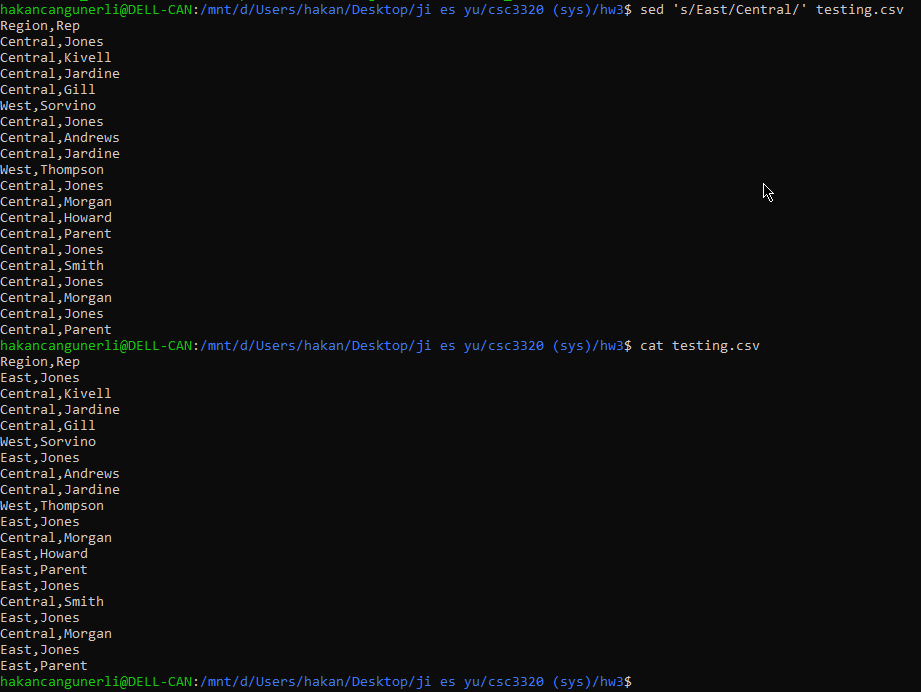
****

1. **sed**

**Command name: sed**

**Usage: stream editing, matching.**

**Output:**

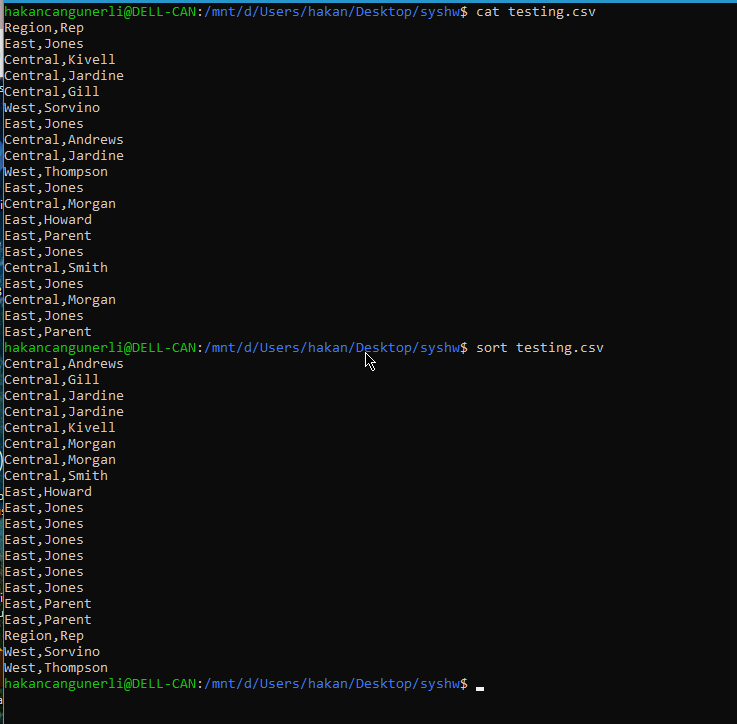
****

1. **sort**

**Command name: sort text files.**

**Usage: sort testing.csv**

**Output:**

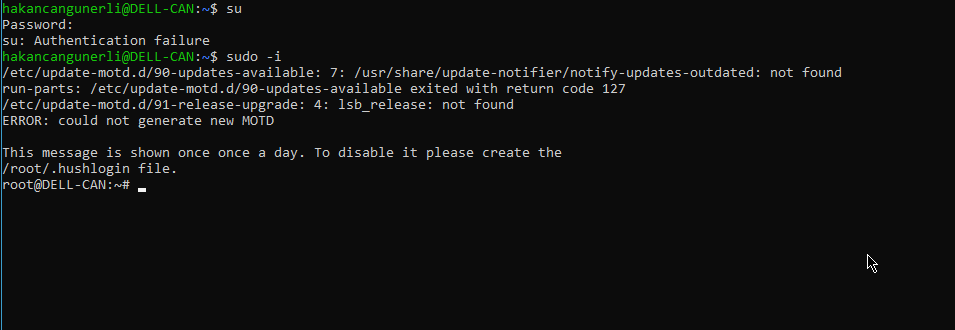


1. **su**

**Command name: su**

**Usage: creates a temporary shell. This won’t work for ubuntu since su is disabled by default.**

**Output**

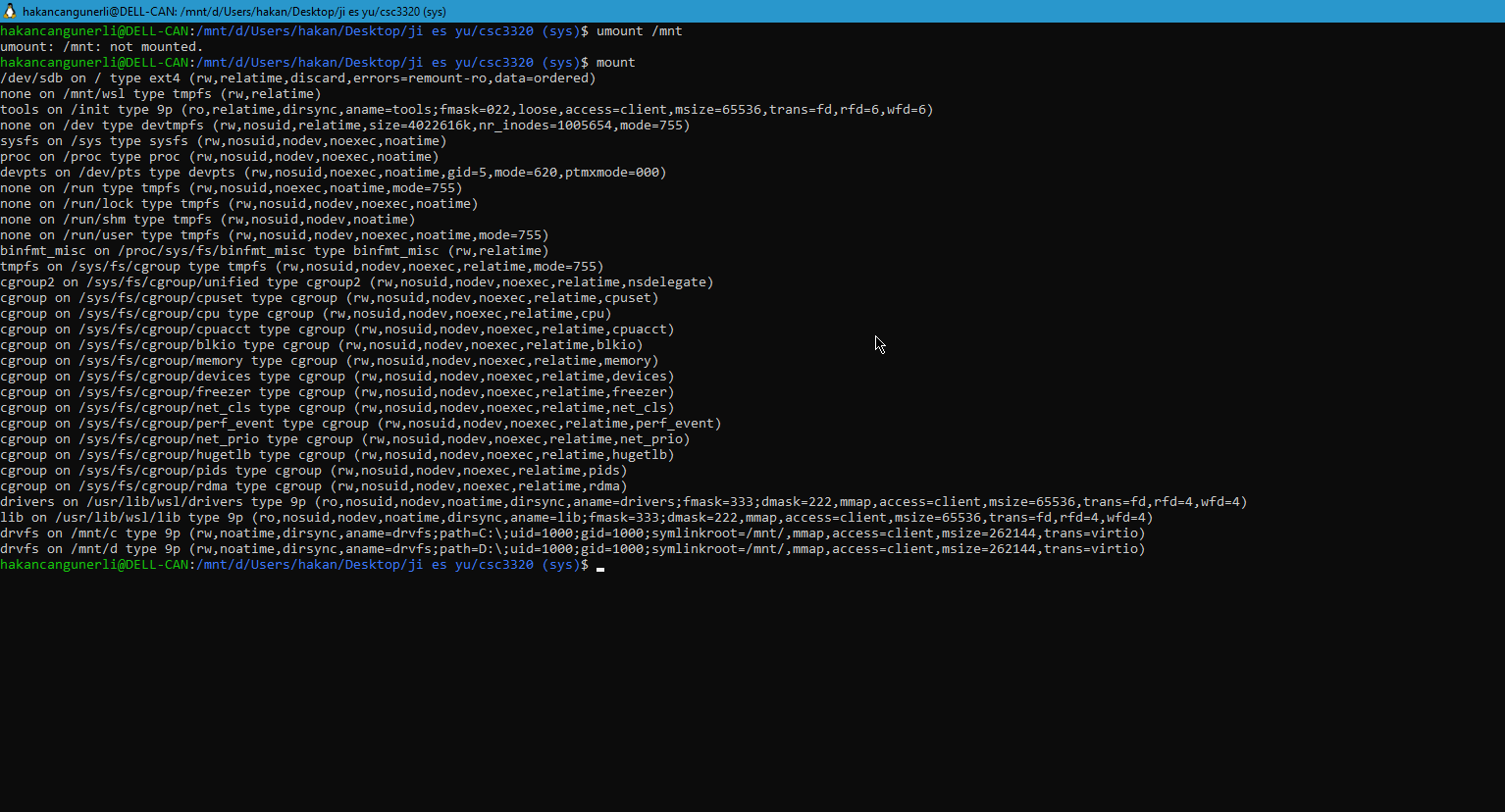
****

1. **umount**

**Command name: umount**

**Usage: unmount a filesystem, antagonist to mount**

**Output**

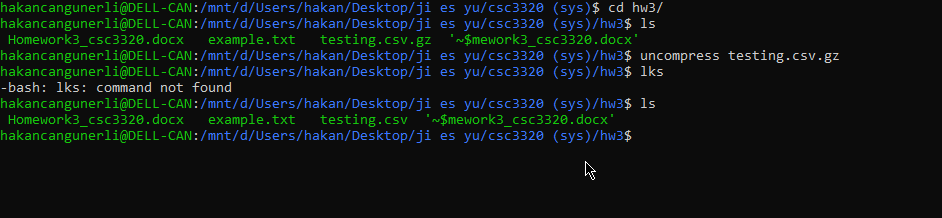
****

1. **uncompress**

**Command name: uncompress**

**Usage: uncompress files, antagonist to compress/gzip.**

**Output**

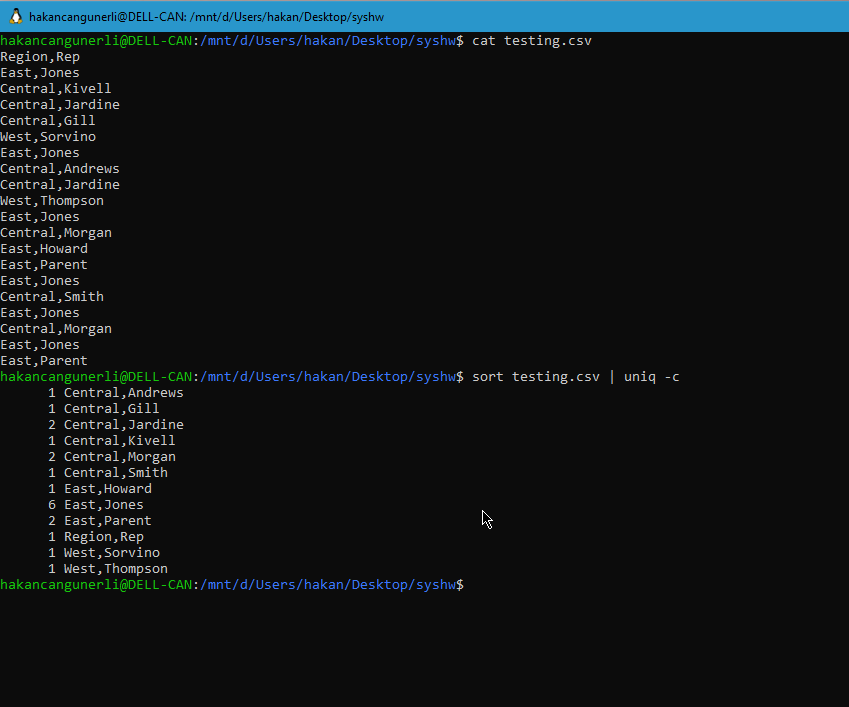
****

1. **uniq**

**Command name: uniq**

**Usage: how many times a character is present uniquely.**

**Output**

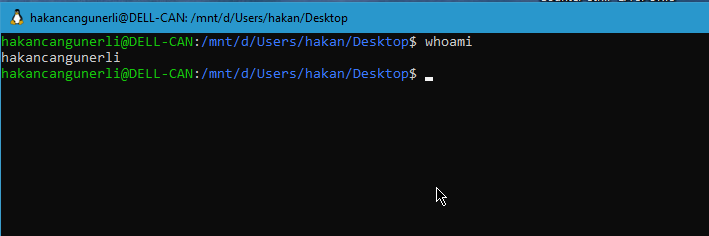


1. **whoami**

**Command name: whoami**

**Usage: print the userid**

**Output**

****

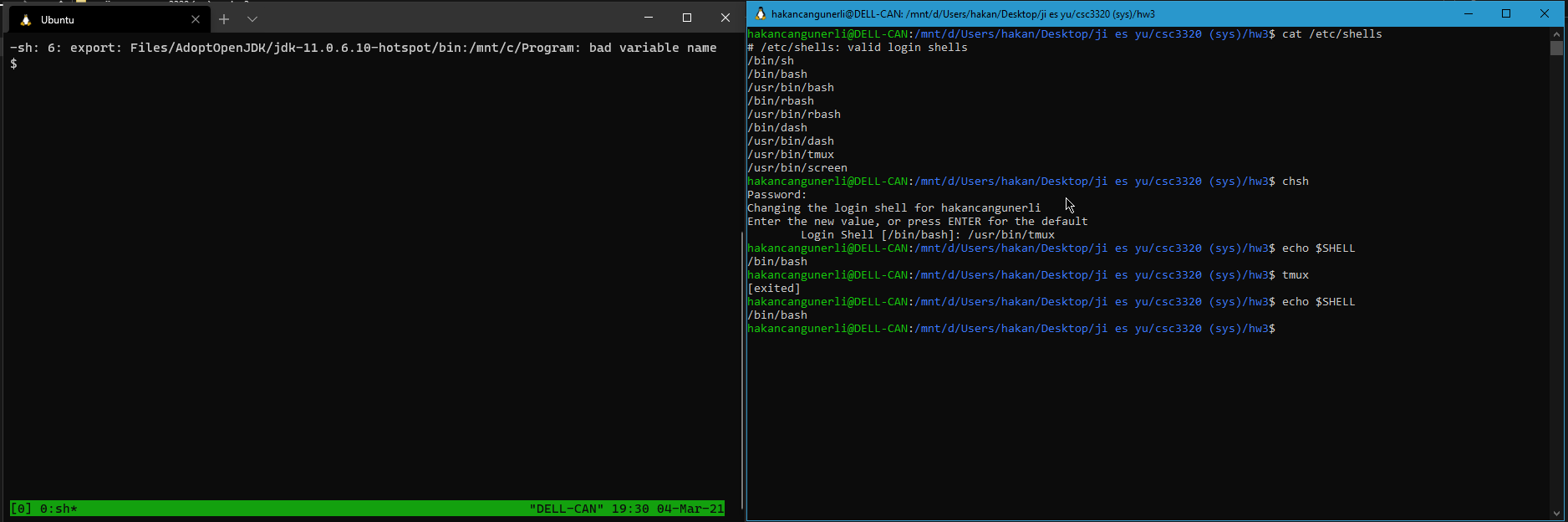
**Part II: Chapter 4**

1. chsh

**Command name: chsh**

**Usage: changing the shell.**

**Output:**

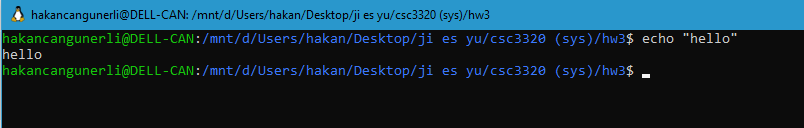


1. echo

**Command name: echo**

**Usage: output a line of text**

**Output:**

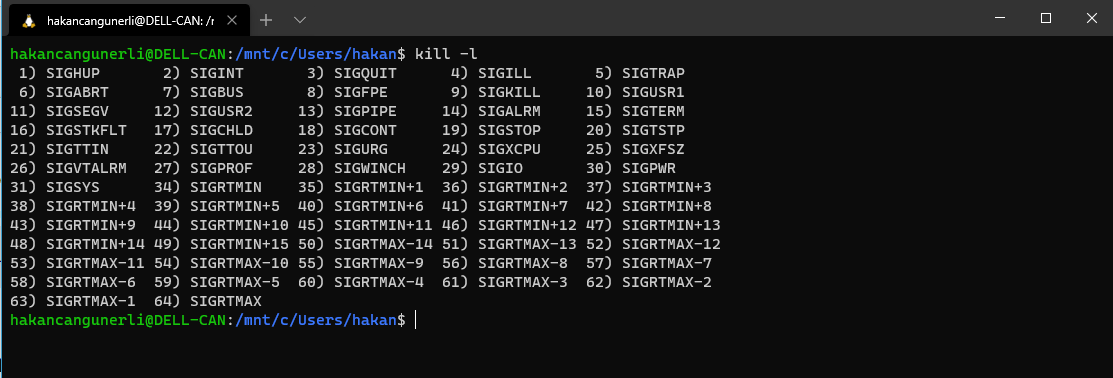


3.kill

**Command name: kill**

**Usage: kill a process**

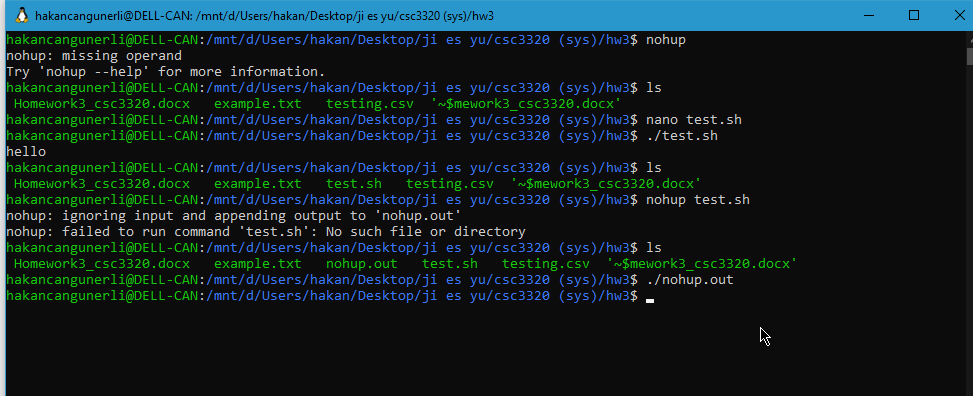
**Output:**

****

1. nohup

**Command name: nohup**

**Usage: run the command even if the session gets disconnected.**

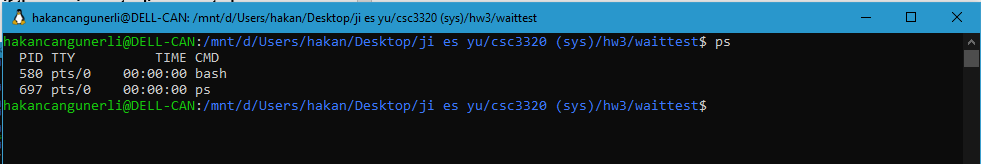
**Output:**

1. ps

**Command name: ps**

**Usage: running processes of a system**

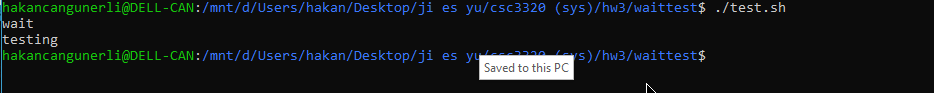
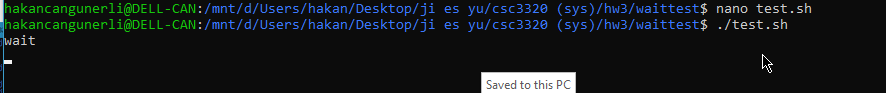
**Output:**



1. sleep

**Command name: sleep**

**Usage: wait for an execution of a script**

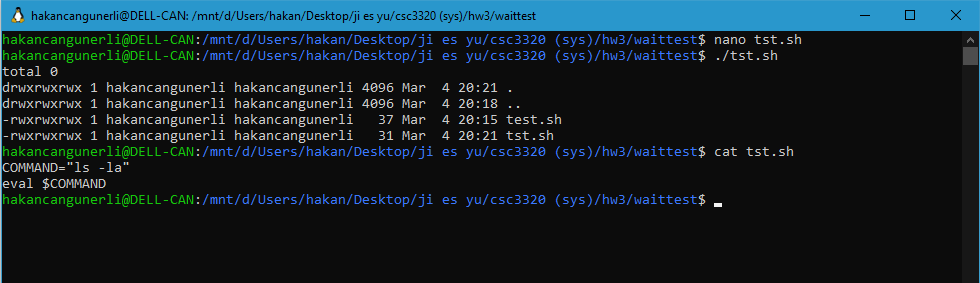
**Output:**

1. eval

**Command name: eval**

**Usage: execute commands, similar to exec.**

**Output:**

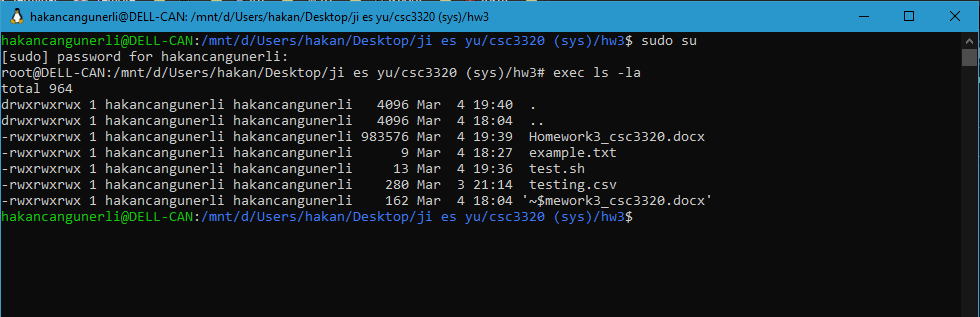


1. exec

**Command name: exec**

**Usage: exec command allows the user to execute a command from the bash.**

**Output:**

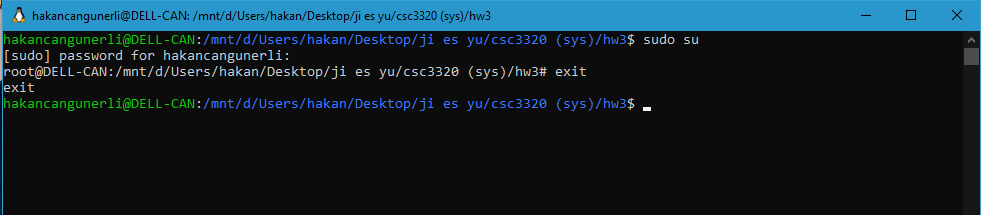


1. exit

**Command name: exit**

**Usage: exit the terminal**

**Output:**

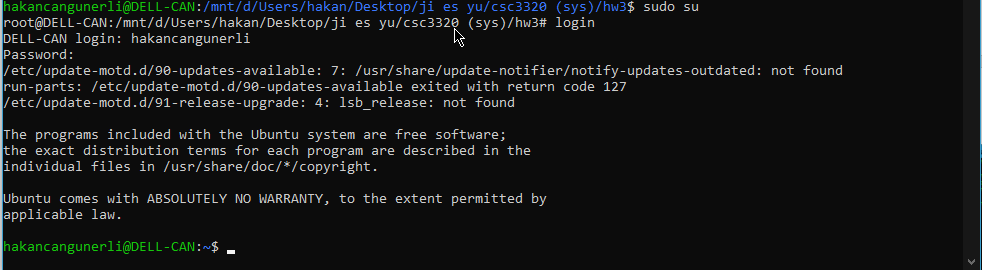


1. login

**Command name: login**

**Usage: login to user**

**Output:**

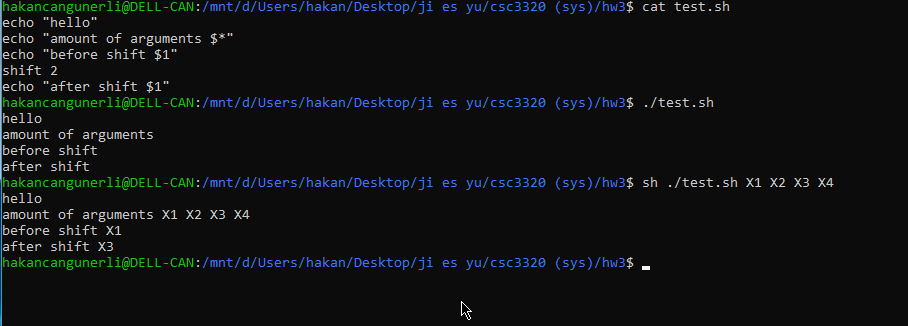


1. shift

**Command name: shift**

**Usage: shift parameters**

**Output:**

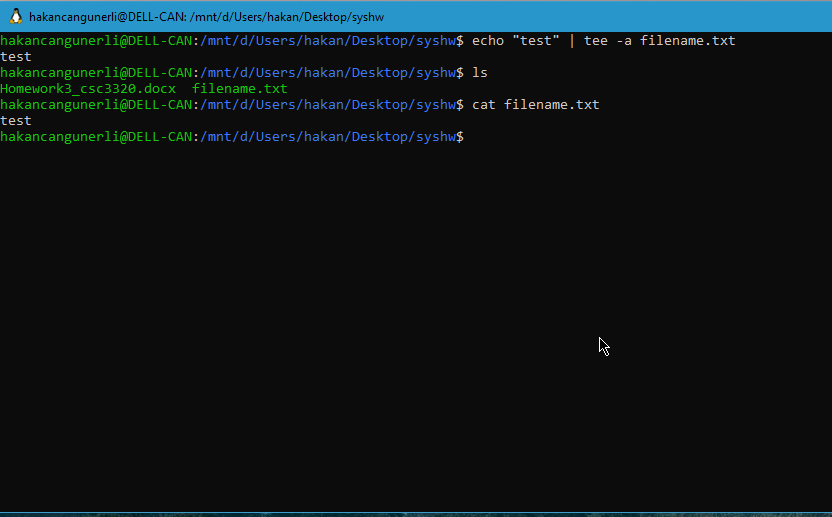


1. tee

**Command name: tee**

**Usage: tee allows to read from the input and write to an output/file.**

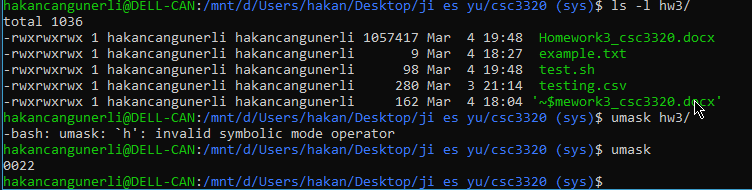
**Output:**



1. umask

**Command name: umask**

**Usage: assign a def file permission for new created files**

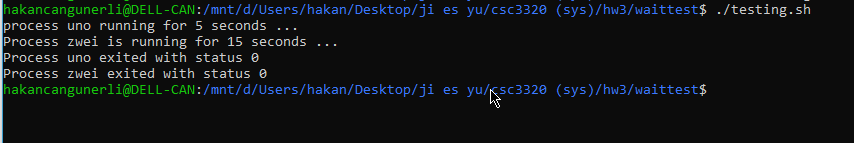
**Output:**

1. wait

**Command name: wait**

**Usage: wait allows you to wait for completing a process.**

**Output:**



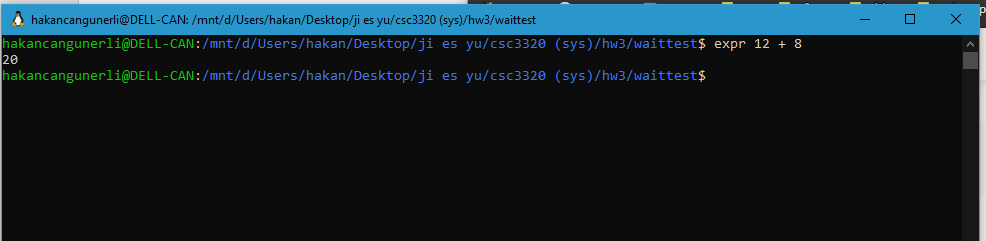
**Part III: Chapter 5**

1.expr

**Command name: expr**

**Usage: evaluate expressions**

**Output:**

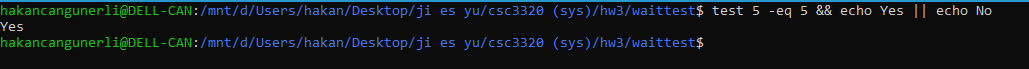
****

2.test

**Command name :test**

**Usage : check and compare values**

**Output:**

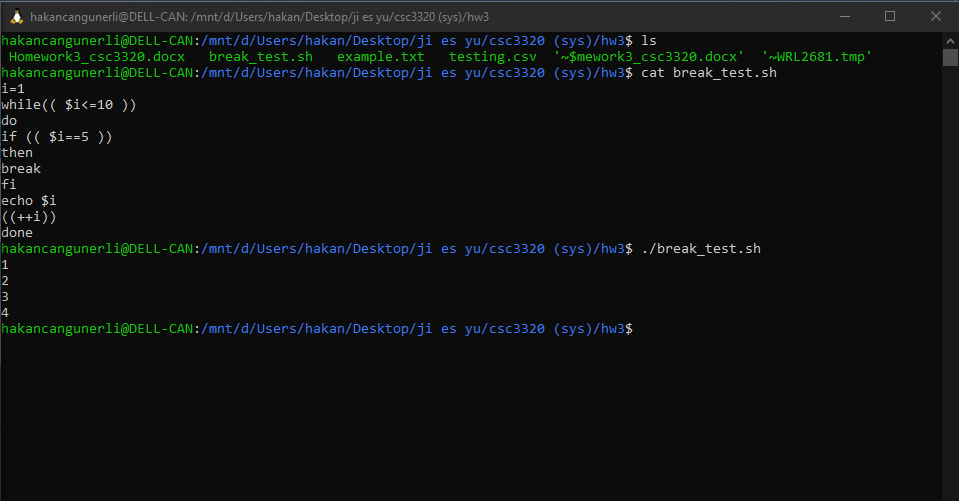


3.break

**Command name :break**

**Usage : terminate the loop**

**Output:**

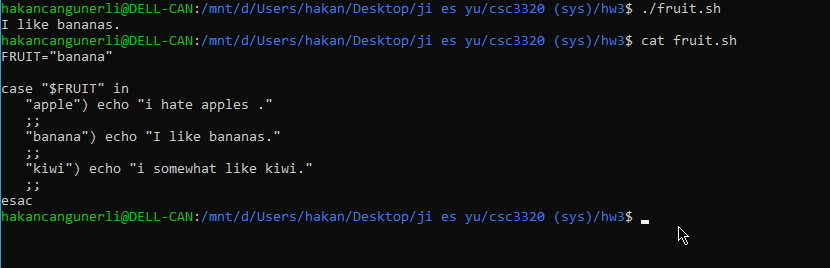


4.case..in..esac

**Command name : case..in..esac**

**Usage : similar to a switch statement**

**Output:**

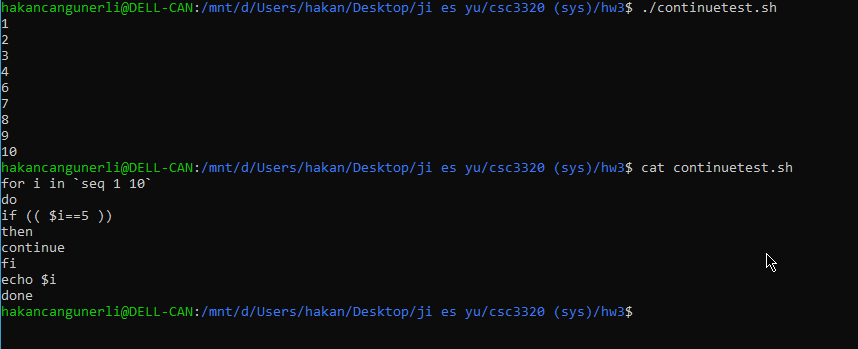


5.continue

**Command name :continue**

**Usage : skip the current iteration for loops.**

**Output:**

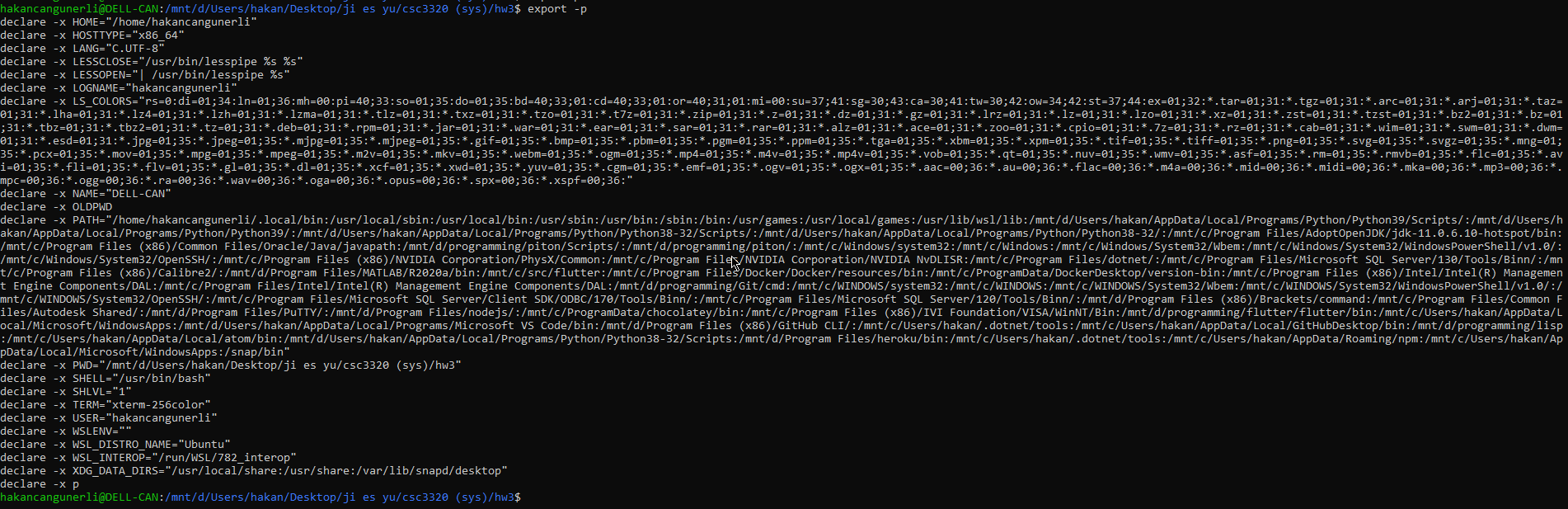


6.export

**Command name : export**

**Usage: export attribute for variables**

**Output:**

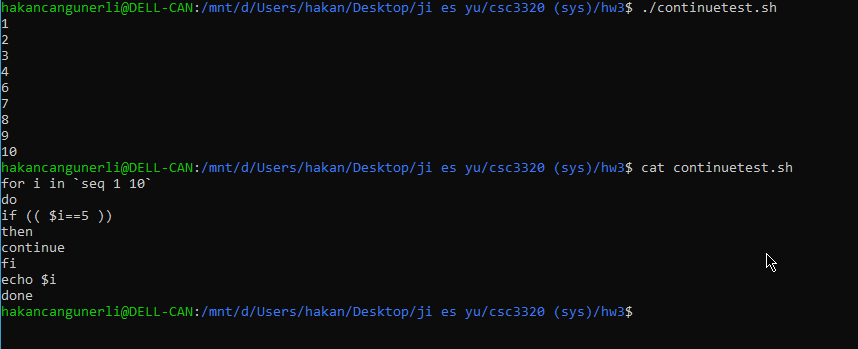


7.for..in..do..done

**Command name : for i in**

**Usage: iteration statement**

**Output:**

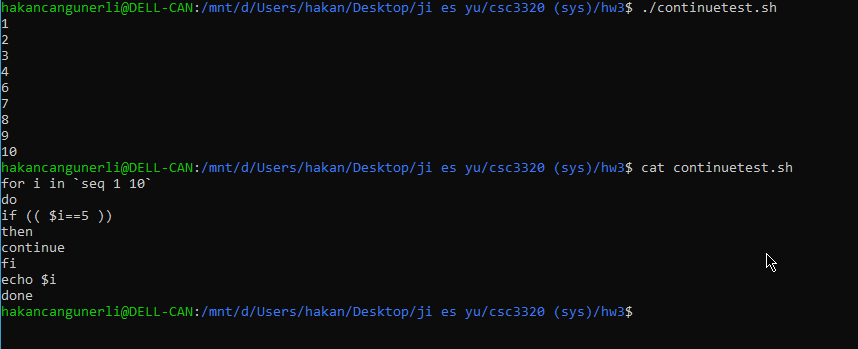


8. if..then..elif..fi

**Command name : if then elif fi**

**Usage: conditional expression**

**Output:**

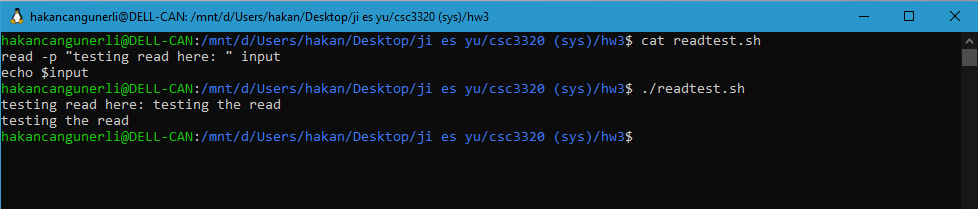


9.read

**Command name : read**

**Usage: read from a file**

**Output:**

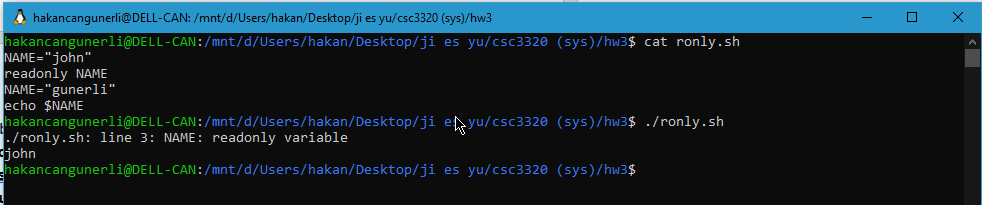


10.readonly

**Command name : readonly**

**Usage: readonly attribute for variables**

**Output:**

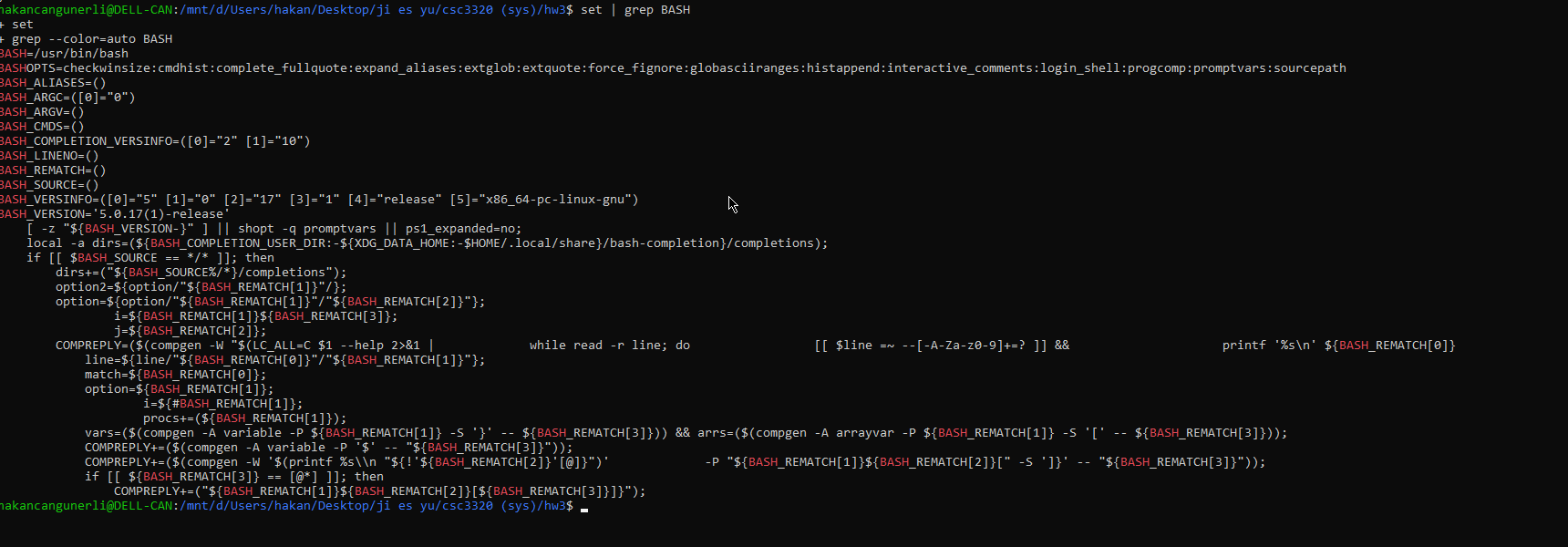


11.set

**Command name : set**

**Usage: set values for shells and parameters.**

**Output:**

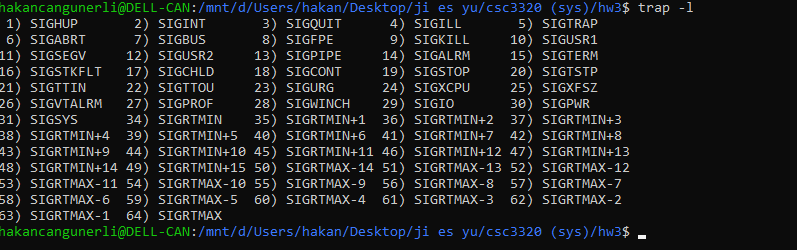
****

12.trap

**Command name : trap**

**Usage: trap signals**

**Output:**



13.while..do..done

**Command name : while..do..done**

**Usage: perform a command multiple times**

**Output:**

