**Birzeit University**

**Computer Science Department**

**Linux OS Laboratory COMP311**

**Lab 5**

**Dr Hafez Barghothi**

**Mohammad Rjoub**

**1220929**

Run the command:

echo PATH

What did you get? **PATH**.

Now run the command:

echo $PATH

What did you get?

**The value of the PATH variable.**

Run the following commands:

SAVEPATH=$PATH

ls Did it work? **Yes**.

PATH=/etc

ls Does it work now? **No**.

Why?

**Because the variable is used by the shell to locate commands for execution is now has a deferent value .**

Restore the original value for variable PATH.

Command? **PATH=$SAVEPATH.**

Now try the command: ls Does it work now? **Yes** .

use the command: PATH=$PATH:/etc

Try it and then use the command: echo $PATH

Was it added as expected? **Yes**.

Display the value of the PWD variable.

Command? **echo $PWD**.

Change your directory to /etc.

Command? **cd /etc**.

What is the value of PWD now? **/etc**.

How do you think the pwd command works? **provides a convenient way for users to check and confirm their current location within the file system hierarchy** .

Now run the following command: PS1=”hello >”

What happened to your prompt? **It changed to hello.**

Now run the command: PS1=’$PWD >’

What happened? **It changed to the value of PWD.**

List three more variables other than the ones mentioned above and their values:

**1-CDPATH** **=' '**

**2-COLORFGBG=** **'15;0'**

**3-COLORTERM=** **truecolor**

Run the command: env | more

Is the output the same as the set command or different? **Different.**

What is the difference between set and env?

**set is used to modify the behavior of the shell, while env is used to set environment variables for a specific command or script without permanently affecting the current shell environment.**

Under your home directory ( cd ) create the following structure:

mkdir project

mkdir project/myfiles

touch project/myfiles/firstfile

Now create a new variable called myprojfilescd as follows:

myprojfiles=$HOME/project/myfiles

Now you can use the new variable to manipulate your project directory.

Try the following commands and write what each does:

vi $myprojfiles/firstfile

**This command opens the firstfile in the project/myfiles directory using the vi text editor.**

cp /etc/passwd $myprojfiles

**This command copies the /etc/passwd file to the project/myfiles directory.**

touch good; mv $HOME/good $myprojfiles

**This command creates an empty file named good in home directory.Then, it moves the good file from home directory to the project/myfiles directory.**

Try the command:

date

What is the result? **The date and the time when the command entered.**

Now try to command:

echo $(date)

What is the result? **The date and the time when the command entered.**

Run the command:

grep yourusername /etc/passwd | cut -d: -f5 | cut -d\_ -f1

What is the result? **My user name.**

Try the new notation to get your last name and save it in a variable called lastname.

Command**: lastname=`grep yourusername /etc/passwd | cut -d: -f5 | cut -d\_ -f1`**

run the command:

alias

List three aliases that you have and their values:

**1-l='ls -CF'**

**2-la='ls -A'**

**3-ll='ls -l'**

Rename (use the mv command) the file .bash\_history to .save\_history.

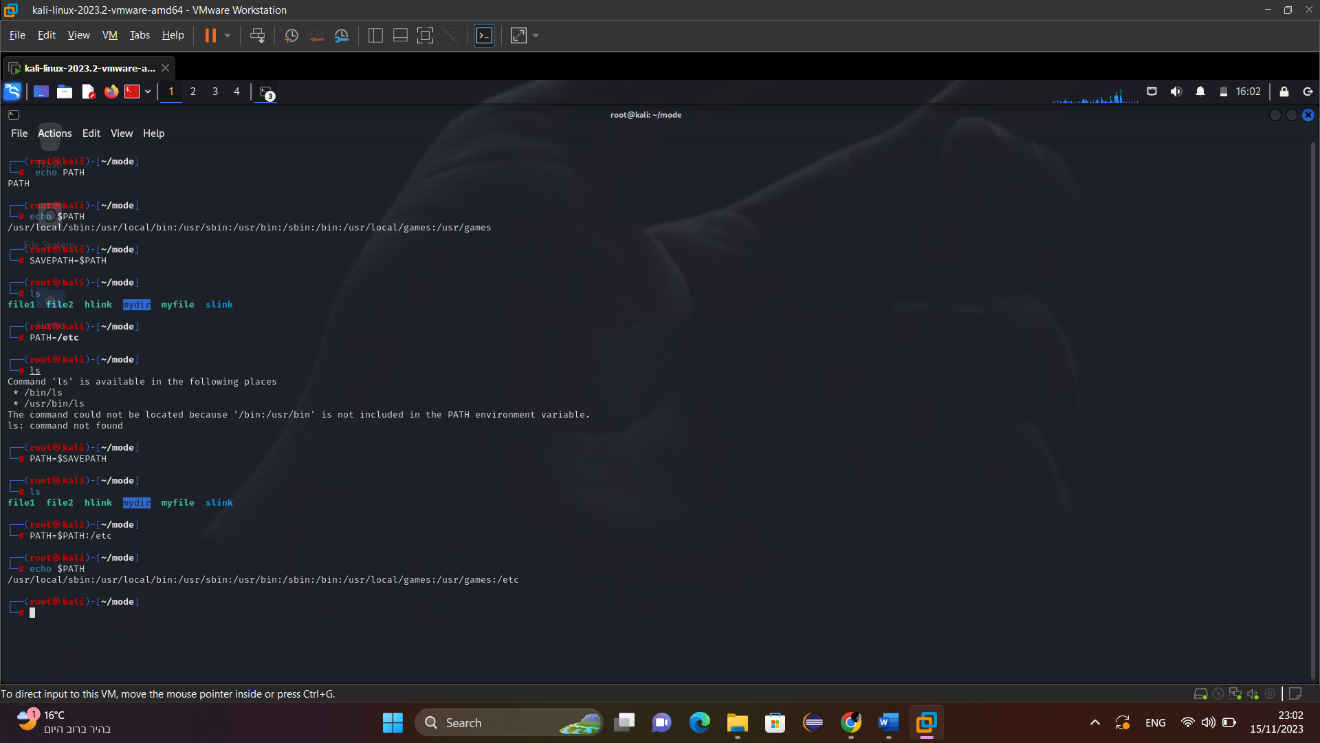
Command: **mv ~/.bash\_history ~/.save\_history**

Exit from the system and log back in. Check the commands stored in .bash\_history. What did you find? Why? **The** **.bash\_history file doesn’t exist because it renamed to .save\_history.**

What can you do to restore all your previous commands? : **mv ~/.save\_history ~/.bash\_history.**

Copy the file .bash\_profile to file .save\_bash\_profile

Command: **cp ~/.bash\_profile ~/.save\_bash\_profile**

****

