Exercise - Users and Groups

Create a user named **dev_user**
Set password of **dev_user** as **clarusway**
Create a group named **dev_team**
Add **dev_user** user to the group **dev_team**
Display **groups** of **dev_user** user
Remove **dev_user** from **dev_team** group
Display **groups** of **dev_user** user
Delete **dev_team** group
Delete **dev_user** user with home directory

sudo sudo
useradd dev-user
etc klasoru icinde kullanicilari gorebiliriz.
cat passwd
passwd dev-user
cat group gruplari grebiliriz.
groupadd dev-team
gpasswd -a dev-user dev-team
groups dev-user
gpasswd -d dev-user dev-team
groups dev-user
groupdel dev-team
userdel -r dev-user # -r home klasorunude siler.

Exercise - Package managers

Update **all** installed packages
Check if **mariadb** is already installed
Find available **mariadb** packages
Install **mariadb**. (Skip confirmations during installation)
List installed **mariadb** package
Uninstall **mariadb** with all unused dependencies
List installed **mariadb** package

sudo yum update -y
sudo yum list installed mariadb
sudo yum list mariadb
sudo yum list installed mariadb
sudo yum autoremove mariadb -y
sudo yum list installed mariadb

Exercise - Control characters

1. a. Search for “clarusway.txt” in the current directory
   b. If it exists display its content
   c. If it does not exist print message “FILE DOES NOT EXIST”
2. Create a file named “clarusway.txt” that contains “Congratulations”
3. Repeat Step 1

```
sudo su
ls clarus*
ls clarusway.txt && more clarusway.txt
cat clarusway.txt
ls clarusway.txt && more clarusway.txt || echo “FILE DOES NOT EXIST”
vim clarusway.txt
i
Congratulations
ESC
:wq
ls clarus*
```

Exercise Bash 1

1. Write a script that;
   a. Asks user to enter two numbers to variables **num1** and **num2.**
   b. Calculates the total of 2 numbers.
   c. Prints the **total** number and applied formula.
2. Make the script executable
3. Add home directory to the PATH
4. Execute the script

```
#!/bin/bash
read -p “Enter a number” num1 num2
let total=$num1 + $num2
print($total)
chmod +x bash1.sh
export PATH=$PATH:/~ # home klasorunu bin altinda olusturdugummuzdan ./ gerek yok
bash1.sh
```

Exercise Bash 2

1. Modify previous script to accept numbers as arguments.

```
#!/bin/bash
Echo “SAyilarin Toplami: $(($1 + $2))
chmod +x bash2.sh
./bash2.sh

bash2.sh 11 22
33
```

Exercise - Filters

1. Create a file named passwords.csv with the following content
   User,Password
   finance,xJ2a_Pl1
   tech,Qc8r7!2P
   hr,l30o_2mM
   operation,12345678
   marketing,qwertyui
   sales,abcdefgh
2. Write a script that accepts user name as argument and prints the password of that user.

```
vim passwords.csv
        User,Password
        finance,xJ2a_Pl1
        tech,Qc8r7!2P
        hr,l30o_2mM
        operation,12345678
        marketing,qwertyui
        sales,abcdefgh

#!/bin/bash

passwd=$(cat password.csv | grep $1 | cut -d’,’ -f2)
echo “password of $1 : $passwd”
chmod +x bash3.sh
./bash3.sh hr
```

Exercise Bash 3

1. Write a script that accepts username as argument
   a. if argument is empty use current user's name
   b. find description of the user and print it
   c. if description is empty print "No description"

2. Create 2 different users with separate descriptions and one without any desciription.
3. Test your script with newly created users.

```
#!/bin/bash
username=$(whoami)
if [ $1 ]
then
        username=$1
fi
description=$(cat /etc/passwd | grep $username | cut -d":" -f5)
if [ "$description" != "" ]
then
echo "Description of $username is $description"
else
echo "No description for user $username"
fi
sudo useradd hr -c “Human Resources Section”
sudo useradd tech -c “Technical Stuff”
sudo useradd aws
**************
#!/bin/bash
username=$(whoami)
if [ $1 ]
then
        username=$1
fi
description=$(cat /etc/passwd | grep $username | cut -d":" -f5)
if [ "$description" != "" ]
then
echo "Description of $username is $description"
else
echo "No description for user $username"
fi
get_password $username    #bu bizim önceki alıştırmada oluşturduğumuz script. burada doğrudan çağırıyoruz.
```

1. Write a script that accepts a path as an argument
   a. If the argument is not empty go to that directory, otherwise stay in the current directory
   b. Add a number from 1 before all the files that has csv extension.

```
if [ $1 ]
then
        cd $1
fi
number=1
for FILE in *.csv
do
        mv $FILE $number-$FILE
        let number++
done
mkdir new
touch test.csv sec.csv third.csv test.txt
```

Exercise Bash 5

Write a script that consists of a function that accepts a directory name as an argument and displays the name of the directory and number of files in the directory.
Name this function “file_count” and call it in your script for;
/etc
~
/usr/bin

```
function file_count()
{
local Directory=$1
COUNT_FILE=$(ls $Directory|wc -l)
echo "$Directory - $COUNT_FILE"
}
file_count /etc
file_count ~
file_count /usr/bin
```