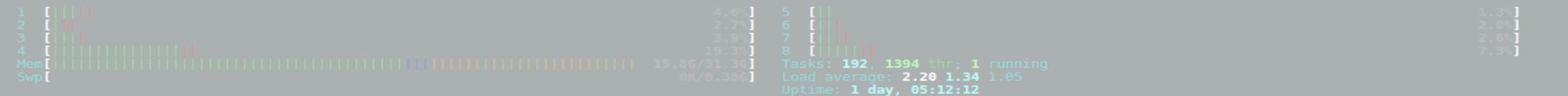


**Nota:** esercitatevi al sito  
<https://overthewire.org/wargames/bandit>



# LINUX: Commandi

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
22108	romano	20	0	2355M	446M	118M	S	20.5	1.4	15:32.05	/usr/lib/firefox/firefox -contentproc -childID 3 -isForBrowser -intPrefs 5:50 6:-1 18:0 28:1000 34:20 35:5 36:10 45:128
25989	romano	20	0	10.3G	8287M	8179M	S	5.9	25.8	41:06.25	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo
4746	root	20	0	367M	189M	79528	S	5.9	0.6	22:33.63	/usr/lib/xorg/Xorg -core :0 -seat seat0 -auth /var/run/lightdm/root/:0 -nolisten tcp vt7 -novtswitch
26015	romano	20	0	10.3G	8287M	8179M	S	4.0	25.8	26:40.40	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo
21198	romano	20	0	2658M	477M	102M	S	2.0	1.5	18:41.80	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
17715	romano	20	0	29252	7044	3244	R	2.0	0.0	0:00.43	htop
20841	romano	20	0	2089M	277M	101M	S	2.0	0.9	8:52.41	/usr/lib/firefox/firefox -contentproc -childID 1 -isForBrowser -intPrefs 5:50 6:-1 18:0 28:1000 34:20 35:5 36:10 45:128
20524	romano	20	0	1727M	408M	161M	S	1.3	1.3	8:07.59	/opt/google/chrome/chrome
24129	romano	20	0	1346M	257M	75564	S	1.3	0.8	7:33.93	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
6829	romano	20	0	1004M	35312	18340	S	1.3	0.1	7:27.57	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo
20665	romano	20	0	2530M	346M	148M	S	0.7	1.1	7:08.98	/usr/lib/firefox/firefox -contentproc -childID 3 -isForBrowser -intPrefs 5:50 6:-1 18:0 28:1000 34:20 35:5 36:10 45:128
20488	romano	20	0	1727M	408M	161M	S	0.7	1.3	20:42.18	/opt/google/chrome/chrome
21453	romano	20	0	1467M	316M	89088	S	0.7	1.0	25:47.84	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
3164	romano	20	0	1377M	256M	73892	S	0.7	0.8	1:55.94	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
25995	romano	20	0	10.3G	8287M	8179M	S	0.7	25.8	3:02.57	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo
21416	romano	20	0	1264M	213M	71824	S	0.7	0.7	2:16.25	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
21144	romano	20	0	2376M	197M	74364	S	0.7	0.6	3:33.32	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
6804	romano	20	0	927M	74232	52148	S	0.7	0.2	4:15.00	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo
4730	root	20	0	204M	13684	8144	S	0.7	0.0	5:43.57	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
8798	romano	20	0	10.3G	8287M	8179M	S	0.7	25.8	0:48.68	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo
6833	romano	20	0	1004M	35312	18340	S	0.7	0.1	3:15.25	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo
20691	romano	20	0	1408M	329M	70840	S	0.7	1.0	1:42.54	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
20741	romano	20	0	1066M	77392	53744	S	0.7	0.2	1:30.00	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo
22108	romano	20	0	2355M	446M	118M	S	0.7	1.4	0:00.00	/usr/lib/firefox/firefox -contentproc -childID 3 -isForBrowser -intPrefs 5:50 6:-1 18:0 28:1000 34:20 35:5 36:10 45:128
20863	romano	20	0	2089M	277M	101M	S	0.7	0.9	0:00.00	/usr/lib/firefox/firefox -contentproc -childID 1 -isForBrowser -intPrefs 5:50 6:-1 18:0 28:1000 34:20 35:5 36:10 45:128
21585	romano	20	0	1207M	138M	65968	S	0.7	0.4	0:00.00	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
22114	romano	20	0	2355M	446M	118M	S	0.7	1.4	0:00.00	/usr/lib/firefox/firefox -contentproc -childID 3 -isForBrowser -intPrefs 5:50 6:-1 18:0 28:1000 34:20 35:5 36:10 45:128
22166	romano	20	0	1429M	347M	74296	S	0.0	1.1	3:02.00	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
17722	romano	20	0	607M	32152	26744	S	0.0	0.1	0:00.00	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo
6823	romano	20	0	163M	12792	9904	S	0.0	0.0	3:03.00	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo
20740	romano	20	0	2530M	346M	148M	S	0.0	1.1	0:40.57	/usr/lib/firefox/firefox
20553	romano	20	0	905M	323M	121M	S	0.0	1.0	10:57.30	/opt/google/chrome/chrome --type=gpu-process --field-trial-handle=12361480615257487612,15625475208483149768,131072 --en
20673	romano	20	0	2530M	346M	148M	S	0.0	1.1	0:31.73	/usr/lib/firefox/firefox
22119	romano	20	0	2355M	446M	118M	S	0.0	1.4	0:32.84	/usr/lib/firefox/firefox -contentproc -childID 3 -isForBrowser -intPrefs 5:50 6:-1 18:0 28:1000 34:20 35:5 36:10 45:128
26023	romano	20	0	10.3G	8287M	8179M	S	0.0	25.8	0:50.54	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo
28988	romano	20	0	2658M	477M	102M	S	0.0	1.5	0:34.15	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
20670	romano	20	0	2530M	346M	148M	S	0.0	1.1	0:35.91	/usr/lib/firefox/firefox
28991	romano	20	0	2658M	477M	102M	S	0.0	1.5	0:34.16	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
22109	romano	20	0	2355M	446M	118M	S	0.0	1.4	0:08.93	/usr/lib/firefox/firefox -contentproc -childID 3 -isForBrowser -intPrefs 5:50 6:-1 18:0 28:1000 34:20 35:5 36:10 45:128
21522	romano	20	0	1241M	151M	67484	S	0.0	0.5	0:26.95	/opt/google/chrome/chrome --type=renderer --field-trial-handle=12361480615257487612,15625475208483149768,131072 --servi
7505	root	20	0	398M	28336	14380	S	0.0	0.1	0:00.94	/usr/lib/snapd/snapd
1047	root	20	0	398M	28336	14380	S	0.0	0.1	0:09.02	/usr/lib/snapd/snapd
20814	romano	20	0	2530M	346M	148M	S	0.0	1.1	0:30.58	/usr/lib/firefox/firefox
26435	romano	20	0	2530M	346M	148M	S	0.0	1.1	0:03.03	/usr/lib/firefox/firefox
22111	romano	20	0	2355M	446M	118M	S	0.0	1.4	0:08.91	/usr/lib/firefox/firefox -contentproc -childID 3 -isForBrowser -intPrefs 5:50 6:-1 18:0 28:1000 34:20 35:5 36:10 45:128
22142	romano	20	0	2355M	446M	118M	S	0.0	1.4	0:08.57	/usr/lib/firefox/firefox -contentproc -childID 3 -isForBrowser -intPrefs 5:50 6:-1 18:0 28:1000 34:20 35:5 36:10 45:128
20857	romano	20	0	2089M	277M	101M	S	0.0	0.9	0:03.34	/usr/lib/firefox/firefox -contentproc -childID 1 -isForBrowser -intPrefs 5:50 6:-1 18:0 28:1000 34:20 35:5 36:10 45:128
26182	romano	20	0	10.3G	8287M	8179M	S	0.0	25.8	2:58.58	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo
29264	romano	20	0	583M	32788	23184	S	0.0	0.1	0:14.77	lxdterminal
5372	romano	20	0	444M	18292	13920	S	0.0	0.1	0:02.68	/usr/bin/lxsession -s Ubuntu -e LXDE
25984	romano	20	0	10.3G	8287M	8179M	S	0.0	25.8	0:44.00	/usr/lib/virtualbox/VirtualBox --comment Windows --startvm 8e348a0f-cf8d-4eab-bdd9-10843800699b --no-startvm-errormsgbo

# Principali comandi Linux

- **Gestione dei file:** ls, cp, mv, rm, touch, head, tail, less, more, find...

```
# mkdir <username>
# cd <username>
# touch miofile
# ls -l miofile
# ls -l > miofile
# less miofile
# head miofile
# cp miofile miofilenew
# rm miofile
```

- **Gestione delle directory:** mkdir, rmdir, mv, cp

```
# mkdir miadir
# cp miofilenew miadir
# cp -av miadir miadirnew
# rm miadir
# rmdir miadir
```

# Principali comandi Linux

- **Gestione delle risorse:** df, du, free, top, htop.

```
# df -h
```

```
# du -h miadirnew
```

```
# free -m
```

```
# top
```

- **Gestione base degli utenti:** adduser, addgroup, useradd, groupadd, usermod, chmod, chown, w, last,....

```
# chmod -R ugo+rwX miadirnew
```

```
# sudo adduser danxiv
```

```
# sudo chown danxiv miadirnew
```

```
# usermod -aG sudo danxiv
```

```
# last; w
```

# Principali comandi Linux

- **Gestione del software:** rpm, yast, urpmi, pacman, emerge, **dpkg**, **apt-get**, **apt-cache**, **apt**.  
# sudo apt install htop  
# htop
- **Spegnimento, riavvio:** shutdown, reboot, poweroff.  
# sudo shutdown -h -t 30

# Principali comandi Linux

- **Gestione dello spazio di memorizzazione di massa locale:** cfdisk, mkfs, mount, parted,

....

```
# mount
```

```
# sudo cfdisk
```

```
# sudo parted
```

- **Configurazione della rete:** ifconfig, route, ip (~~/etc/network/interfaces,~~  
~~/etc/resolv.conf~~)

```
# ifconfig -v; route;
```

```
# cat /etc/netplan/*.yaml
```

```
# cat /etc/resolv.conf
```

- ~~**Controllo delle operazioni periodiche:** cron, at (/etc/crontab) e /etc/cron.d,...~~

```
# at now
```

# Esempio configurazione netplan

```
network:
```

```
  version: 2
```

```
  renderer: networkd
```

```
  ethernets:
```

```
    eth1:
```

```
      addresses:
```

```
        - 10.10.10.2/24
```

```
      gateway4: 10.10.10.1
```

```
      nameservers:
```

```
        search: [mydomain, otherdomain]
```

```
        addresses: [10.10.10.1, 1.1.1.1]
```

# Principali comandi Linux

- **whoami**: ottenere informazioni sul proprio username.
- **w**: ci dice che è collegato al sistema.
- **finger**: da informazioni dettagliate su un utente.

```
# finger danxiv
```

- **id**: restituisce l'uid dell'utente e dei gruppi di cui fa parte:

```
# id danxiv
```

- Col comando **groups** è possibile vedere di quali gruppi fa parte un utente. Ad esempio:

```
# groups danxiv
```



# Principali comandi Linux

- Per vedere la **lista dei processi attivi** si usa (di solito con vari parametri) il comando **ps** (es: `ps aux`).
- È inoltre possibile vedere la **genealogia dei processi** (chi ha lanciato chi) tramite il comando **pstree**.
- Una visione più dinamica dei processi in esecuzione si ha tramite il comando **top**, o la sua alternativa **htop**.
- È possibile mandare un **segnale** o **uccidere** un processo tramite il comando **kill** o **killall** seguito dal numero di segnale e dal **pid** (o **nome**) del processo:

```
# kill -9 2113
```

```
# killall -9 firefox
```

```
# kill `pidof chrome`
```

- È inoltre possibile modificare la **priorità di esecuzione** di un processo usando i comandi:
  - **nice**: lancia un programma con una determinata priorità.
  - **renice**: cambia la priorità di un processo in esecuzione.
  - **ionice**: cambia la classe di I/O scheduling (*idle*, *realtime*, *best-effort*) e la priorità del processo.

# Profilazione degli utenti amministratore

- Se più sistemisti si occupano dello stesso sistema è bene che vi sia un utente amministratore per ognuno di essi.
- Create quindi un nuovo utente nel vostro sistema tramite il comando `adduser`, ad esempio:

```
# adduser <nomecomponentedelgruppo>_adm
```

- Date al nuovo utente i permessi di root tramite:

```
# usermod -aG sudo <nomecomponentedelgruppo>_adm
```

- Provate nuovo utente!!

Entrate nel sistema come utente

```
<nomecomponentedelgruppo>_adm.
```

1) Provate ad eseguire un comando con `sudo`.

# Powershell

- Ecco alcuni esempi presi da:  
<https://docs.microsoft.com/it-it/powershell/scripting/samples/collecting-information-about-computers?view=powershell-7>

# Powershell - Esempi

- Elenco delle informazioni sul BIOS

```
Get-CimInstance -ClassName Win32_BIOS
```

- Elenco delle informazioni sul processore

```
Get-CimInstance -ClassName Win32_Processor  
| Select-Object -ExcludeProperty "CIM*"
```

- Elenco degli hotfix installati

```
Get-CimInstance -ClassName  
Win32_QuickFixEngineering
```

# Powershell - Esempi

- Elenco di informazioni sulla versione del sistema operativo

```
Get-CimInstance -ClassName  
Win32_OperatingSystem |  
Select-Object -Property  
BuildNumber,BuildType,OSType,ServicePackMa  
jorVersion,ServicePackMinorVersion
```

# Powershell - Esempi

- Elenco di utenti locali e proprietario

```
Get-CimInstance -ClassName  
Win32_OperatingSystem | Select-Object -  
Property *user*
```

- Recupero dello spazio su disco disponibile

```
Get-CimInstance -ClassName  
Win32_LogicalDisk -Filter "DriveType=3"
```

**BASH-**


**ESERCIZI**

```
~root: env X="( echo; ) && echo completed" /bin/sh -c echo completed"  
> shellshock  
> completed
```

# Hello World

- Direttamente dal prompt dei comandi:

```
# echo "Hello word!"
```



```
ubuntu@ubuntu-VirtualBox:~$ echo "Hello World"  
Hello World  
ubuntu@ubuntu-VirtualBox:~$
```

—

- PS: non mi sono inventato nulla, esempi presi da:  
[https://linuxhint.com/30\\_bash\\_script\\_examples/](https://linuxhint.com/30_bash_script_examples/)



# Creare uno script

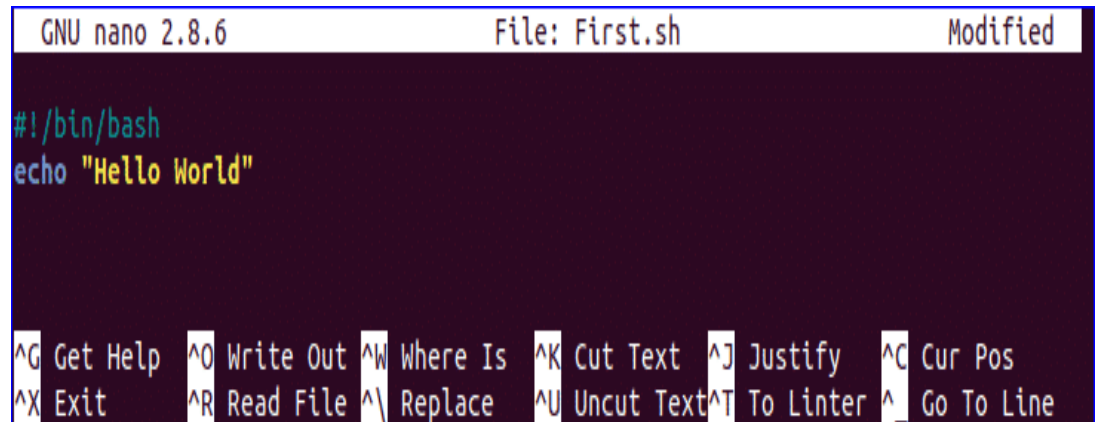
- Utilizzare un editor di testo per creare un file contenente il vostro script:

```
# nano hw.sh
```

- Inserire il contenuto dello script:

```
#!/bin/bash
```

```
echo "Hello world!"
```



```
GNU nano 2.8.6                                File: First.sh                                Modified
```

```
#!/bin/bash  
echo "Hello World"
```

```
^G Get Help  ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos  
^X Exit      ^R Read File  ^\ Replace    ^U Uncut Text ^T To Linter  ^_ Go To Line
```

# Creare uno script

- Per eseguire lo script ci sono due modi:
  - Chiamata tramite shell:  
`# bash hw.sh`
  - Assegnando i permessi di esecuzione al file:  
`# chmod +x hw.sh`  
`# ./hw.sh`

```
ubuntu@ubuntu-VirtualBox:~$ bash First.sh
Hello World
ubuntu@ubuntu-VirtualBox:~$ chmod a+x First.sh
ubuntu@ubuntu-VirtualBox:~$ ./First.sh
Hello World
ubuntu@ubuntu-VirtualBox:~$
```

# Stampare messaggi a video

- Per stampare messaggi a video utilizziamo il comando **echo** (salvate il file come `echo.sh`):

```
#!/bin/bash
```

```
echo "Testo con newline"
```

```
echo -n "Testo senza newline"
```

```
echo -e "\n Testo \t senza \t backslash \n"
```

```
ubuntu@ubuntu-VirtualBox:~/code$ bash echo_example.sh
```

```
Printing text with newline
```

```
Printing text without newline
```

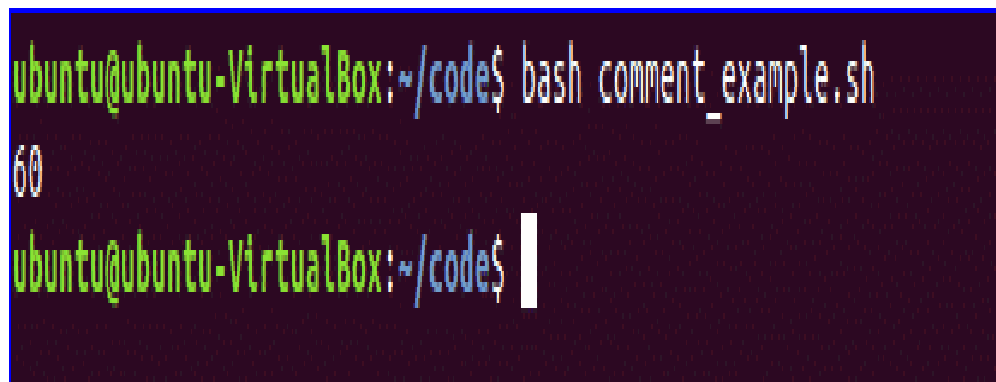
```
Removing          backslash          characters
```

```
ubuntu@ubuntu-VirtualBox:~/code$
```

# Uso dei commenti e espressioni algebriche

- Questo script, `commenti.sh` mostra l'uso dei commenti e effettua la somma di due numeri:

```
#!/bin/bash
# Somma due numeri a piacere
((sum=25+35))
#Stampa il risultato
echo $sum
```

A terminal window with a dark background and green text. The prompt is 'ubuntu@ubuntu-VirtualBox:~/code\$'. The user enters 'bash comment\_example.sh'. The output is '60'. The prompt returns to 'ubuntu@ubuntu-VirtualBox:~/code\$' with a white cursor at the end.

```
ubuntu@ubuntu-VirtualBox:~/code$ bash comment_example.sh
60
ubuntu@ubuntu-VirtualBox:~/code$
```

- Esecuzione:

```
# chmod +x commenti.sh && ./commenti.sh
```

# Ciclo WHILE

- Incrementiamo una variabile partendo da 1 fino ad 5 (salvate il file come `while.sh`):

```
#!/bin/bash
```

```
valid=true
```

```
count=1
```

```
while [ $valid ];do
```

```
    echo $count
```

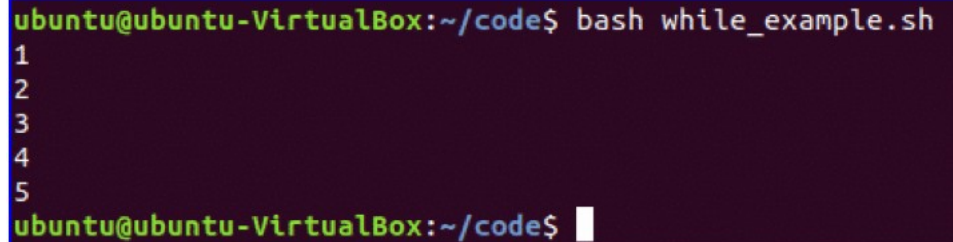
```
    if [ $count -eq 5 ]; then
```

```
        break
```

```
    fi
```

```
    ((count++));
```

```
done
```

A terminal window with a dark background and light green text. The prompt is 'ubuntu@ubuntu-VirtualBox:~/code\$'. The user has entered 'bash while\_example.sh'. The output shows the numbers 1, 2, 3, 4, and 5, each on a new line. The prompt is now 'ubuntu@ubuntu-VirtualBox:~/code\$' with a cursor at the end.

```
ubuntu@ubuntu-VirtualBox:~/code$ bash while_example.sh
1
2
3
4
5
ubuntu@ubuntu-VirtualBox:~/code$
```

# Ciclo FOR

- Stampiamo il conto alla rovescia per il lancio di un missile (salvate il file come `for.sh`):

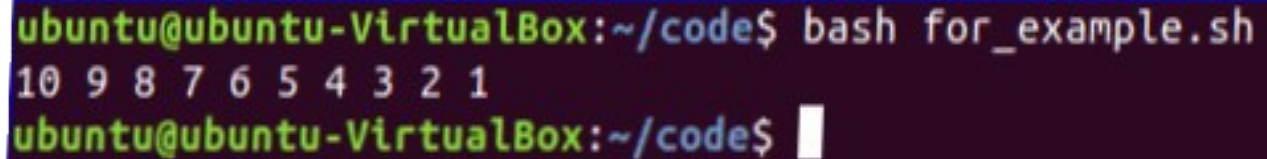
```
#!/bin/bash
```

```
for ( ( counter=10; counter>0; counter-- ) ) ; do
```

```
    echo -n "$counter "
```

```
done
```

```
echo ""
```

A terminal window with a dark background. The prompt is 'ubuntu@ubuntu-VirtualBox:~/code\$'. The user has entered 'bash for\_example.sh'. The output is '10 9 8 7 6 5 4 3 2 1'. The prompt is now 'ubuntu@ubuntu-VirtualBox:~/code\$' with a cursor.

```
ubuntu@ubuntu-VirtualBox:~/code$ bash for_example.sh
10 9 8 7 6 5 4 3 2 1
ubuntu@ubuntu-VirtualBox:~/code$
```

# Costrutto IF

- Un blocco IF inizia con la parola chiave **if** e termina con **fi**. La clausola è racchiusa fra parentesi quadre (salvate come `if.sh`).

```
#!/bin/bash
```

```
# Leggo n da tastiera
```

```
read n
```

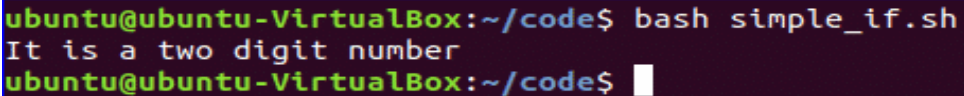
```
if [ $n -lt 10 ]; then
```

```
    echo "Numero ad una sola cifra"
```

```
else
```

```
    echo "Numero a due o più cifre"
```

```
fi
```

A terminal window with a dark purple background. The prompt is 'ubuntu@ubuntu-VirtualBox:~/code\$'. The user has entered 'bash simple\_if.sh'. The output is 'It is a two digit number'. The prompt is now 'ubuntu@ubuntu-VirtualBox:~/code\$' with a cursor.

```
ubuntu@ubuntu-VirtualBox:~/code$ bash simple_if.sh
It is a two digit number
ubuntu@ubuntu-VirtualBox:~/code$
```

# Costrutto IF

- Le condizioni posso essere anche risultato di espressioni logiche con **and** e **or**:

```
#!/bin/bash
```

```
echo "Login:"
```

```
read username
```

```
echo "Password:"
```

```
read password
```

```
if [[ ( $username == "admin" && $password == "secret" ) ]]; then
```

```
    echo "valid user"
```

```
else
```

```
    echo "invalid user"
```

```
fi
```



# Costrutto IF

- Le condizioni possono essere anche risultato di espressioni logiche con **and** e **or**:

```
#!/bin/bash
```

```
echo "Enter any number"
```

```
read n
```

```
if [[ ( $n -eq 15 || $n -eq 45 ) ]]; then
```

```
    echo "You won the game"
```

```
else
```

```
    echo "You lost the game"
```

```
fi
```

# Costrutto IF

- La clausola **else if (elif)** permette di annidare più costrutti if in un unico blocco:

```
#!/bin/bash
```

```
echo "Enter your lucky number"
```

```
read n
```

```
if [ $n -eq 101 ];then
```

```
    echo "You got 1st prize"
```

```
elif [ $n -eq 510 ];then
```

```
    echo "You got 2nd prize"
```

```
elif [ $n -eq 999 ];then
```

```
    echo "You got 3rd prize"
```

```
else
```

```
    echo "Sorry, try for the next time"
```

```
fi
```

```
ubuntu@ubuntu-VirtualBox:~/code$ bash elseif_example.sh
Enter your lucky number
101
You got 1st prize
ubuntu@ubuntu-VirtualBox:~/code$ bash elseif_example.sh
Enter your lucky number
999
You got 3rd prize
ubuntu@ubuntu-VirtualBox:~/code$ bash elseif_example.sh
Enter your lucky number
100
Sorry, try for the next time
ubuntu@ubuntu-VirtualBox:~/code$
```

# Costrutto CASE

- In alternativa a else if è possibile usare il costrutto **case**, il blocco inizia con la clausola **case** e si chiude con la clausola **esac**, i casi sono evidenziati da parentesi tonda (salvate il file come `case.sh`):

```
#!/bin/bash
echo "Enter your lucky number"
read n
case $n in
    101)echo echo "You got 1st prize" ;;
    510)echo "You got 2nd prize" ;;
    999)echo "You got 3rd prize" ;;
    *)echo "Sorry, try for the next time" ;;
esac
```

# Funzioni

- La Bash non permette la creazioni di funzioni con parametri, ma è possibile utilizzare dei parametri sfruttando altre variabili(salvate il file come `func.sh`):

```
#!/bin/bash
```

```
Area_Rettangolo() {  
    area=$(( $1 * $2 ))  
    echo "Area is : $area"  
}
```

```
Area_Rettangolo 10 20
```

# Funzioni

- Si possono passare parametri alle funzioni sia come numeri che come stringhe:

```
#!/bin/bash
```

```
Area_Quadrato() {  
    area=$((lato * lato))  
}
```

```
Echo "Inserisci il lato:"
```

```
read lato
```

```
Area_Quadrato
```

```
echo "Area: $area"
```

# Funzioni

- In realtà le funzioni accettano parametri di input come argomenti similmente agli script e restituiscono i valori come stati come se fossero comandi:

```
#!/bin/bash
```

```
Area_Quadrato() {
```

```
    area=$(( $1 * $1 ))           # $1 parametro di input
```

```
    return $area
```

```
}
```

```
Echo "Inserisci il lato:"
```

```
read lato
```

```
Area_Quadrato $lato
```

```
a=$?                               # ritorna il risultato
```

```
dell'esecuzione dell'ultimo comando
```

```
echo "Area: $a"
```

# Esistenza di file/directory

- Tramite il costrutto if e degli switch particolari è possibile testare l'esistenza di file o directory:

```
#!/bin/bash
filename=$1
if [ -f "$filename" ]; then
    echo "File exists"
else
    echo "File does not exist"
fi
dirname=$2
if [ -d "$dirname" ]; then
    echo "Dir exists"
else
    echo "Dir does not exist"
fi
```

## Altri esempi

- Provate a cambiare la prima riga di uno dei vostri script precedenti in questo modo:
- `#!/bin/bash -x`
- `#!/bin/bash -e`

Cosa succede?



# Altri esempi

- Creare uno script che aggiorni data e ora ogni secondo:

```
#!/bin/bash
While [ 1 = 1 ];
do
    date
    sleep 1
done
```

- Contare il numero di file di log di un sistema:

```
#!/bin/bash
```

```
ls -l /var/log/*.log | wc -l
```

- Cercare le attività di un utente nei log:

```
#!/bin/bash
```

```
# $1 contiene lo username.
```

```
cat /var/log/*.log | grep $1
```

# Altri esempi

- Creare uno script estragga nome utente e gecos dal file `/etc/passwd`:

```
#!/bin/bash
```

```
getent passwd | cut -d: -f1,5
```

- Estendere lo script precedente per creare delle directory per tali utenti:

```
#!/bin/bash
```

```
for i in $(getent passwd | cut -d: -f1); do
```

```
    mkdir /scratch/$i
```

```
    chown -R $i /scratch/$i
```

```
done;
```

## Altri esempi

- Realizzare uno script che stampa il contenuto di una directory con il peso di ogni file:

```
#!/bin/bash  
  
for i in $(ls -l); do  
    du -hs $i;  
done;
```

# Esempio di until..do

```
x=100
until [ "$x" -ge 10 ]; do
    echo "Valore corrente di x: $x";
    x=$(( $x - 1 ))
done
```

# Altri esempi

- Realizzare uno script che stampa solo i file pdf presenti in una directory:

```
#!/bin/bash
```

```
for i in $(ls -l *.pdf) ; do
```

```
    lpr $i;
```

```
done;
```

# Esempio Backup propria Home

- Realizzare uno script che faccia un backup della vostra home:

```
#!/bin/bash
```

```
tar cvf $USER.tar $HOME
```

- Realizzare uno script che faccia un backup della vostra home con data e ora:

```
#!/bin/bash
```

```
tar cvf $USER-$(date '+%d-%m-%Y_%H:%m') .tar $HOME
```

# Esempio Backup Home Utenti

- Supponendo di essere root di un sistema in cui tutti gli utenti hanno home in /home, creare uno script che effettua il backup della home di ogni singolo utente in /backup/nomeutente:

```
#!/bin/bash

mkdir -p /backup

for i in $(ls -1 /home); do
    mkdir -p /backup/$i;
    tar cvf /backup/$i/$i-$(date '+%d-%m-%Y').tar
    /home/$i;
done;
```



# Esempio cancellazione e backup utenti

- Realizzare uno script che, a partire da un file con un elenco di utenti elimini le home di tali utenti residenti in `/home` dopo un backup:

```
#!/bin/bash

mkdir -p /backup

for i in $(cat utentidaeliminare.txt) ; do
    echo Cancellato $i
    tar zcf /backup/$i.tgz /home/$i
    rm -rf /home/$i
    echo Home $i eliminata
done;
```

# Esempio case

Realizzare uno script che prende in input una lettera ed esegue il comando corrispondente `date`, `who`, `ls -l | more`, `pwd`:

```
#!/bin/bash
```

```
# scelte di comandi
```

```
case $1 in
```

```
    'D') date;;
```

```
    'W') who;;
```

```
    'L') ls -l|more;;
```

```
    'P') pwd;;
```

```
    *)
```

```
        echo 'D = data odierna'
```

```
        echo 'W = utenti collegati'
```

```
        echo 'L = lista dei file'
```

```
        echo 'P = directory corrente';;
```

```
esac
```

# Idea di progetto

- Utilizzando i cicli, le funzioni, il costrutto case e i comandi di sistema Linux, realizzare uno script in bash che permette una semplice gestione di un server linux:
  - Gestione degli utenti
  - Gestione dei demoni
  - Gestione del file system
  - Gestione della rete
  - Ecc ecc