

ESERCITAZIONE W5D2

In questo esercizio abbiamo usato alcuni dei comandi fondamentali della Shell Linux per navigare, creare cartelle, copiare file, modificare permessi e gestire il filesystem. Il tutto partendo dalla HOME e facendo tutte le operazioni con istruzioni testuali. Premetto con il dire che all'inizio ho sbagliato il posizionamento iniziale delle cartelle, e all'inizio ho creato un documento nella cartella sbagliata, appena visto notato gli errori li ho corretti rispettivamente in *figura 6* e *figura 7*. L'immagine seguente ci mostra la situazione di partenza dalla quale partire.

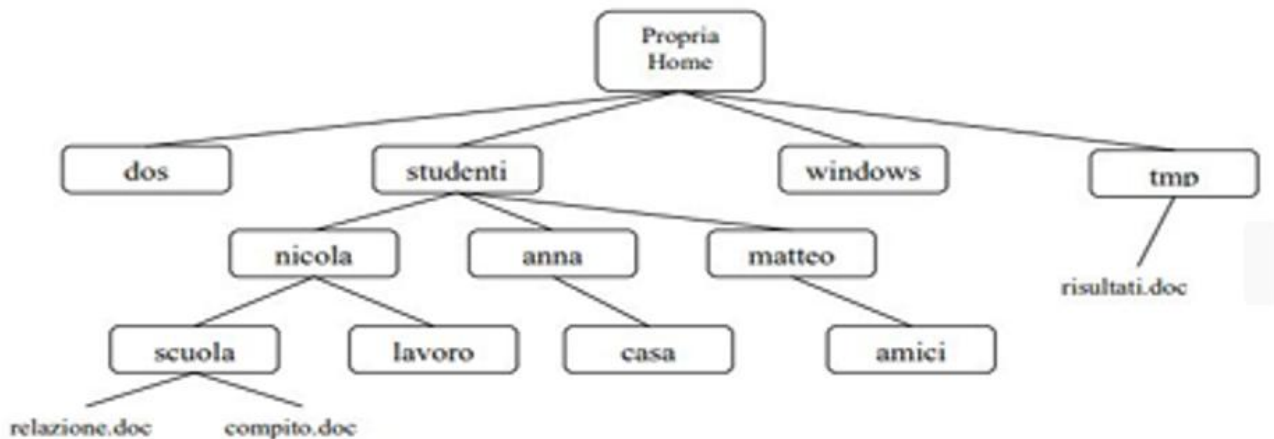


Figura 0

Partendo dall'inizio creiamo le cartelle con il comando "mkdir" dove iniziamo a creare le prime cartelle "dos", "studenti", "windows" e "tmp".

```
kali@kali: ~
$ mkdir dos
$ mkdir student
$ mkdir windows
$ mkdir tmp
$ ls
Desktop  Documents  dos  Downloads  Music  Pictures  Public  ssl-test  student  Templates  tmp  Videos  windows
```

Lo screenshot mostra un terminale Kali Linux all'interno di una macchina virtuale Oracle VM VirtualBox. Sono stati eseguiti i comandi per creare le directory "dos", "student", "windows" e "tmp". Il prompt di comando è attualmente "kali@kali: ~".

Figura 1

Poi iniziamo a creare le prime sottocartelle "anna" e "casa" che vanno nella cartella principale "studenti". Questa volta però per accelerare un po' il processo usiamo il comando "mkdir -p" e per rendere di più semplice lettura l'immagine lo visualizziamo con il comando "tree"

```
(kali㉿kali)-[~]
$ mkdir -p studenti/anna/casa/

(kali㉿kali)-[~]
$ tree
.
├── Desktop
├── Documents
├── dos
├── Downloads
├── Music
├── Pictures
├── Public
├── ssl-test
│   ├── cert.pem
│   ├── index.html
│   └── key.pem
├── student
├── studenti
│   ├── anna
│   │   └── casa
├── Templates
├── tmp
├── Videos
└── windows

17 directories, 3 files
```

Figura 2

Nell'immagine seguente usiamo il comando "cd" per andare nella cartella indicata per poi creare le altre due sottocartelle allo stesso livello di "anna" e le chiameremo "Nicola" e "Matteo"

```
(kali㉿kali)-[~]
$ cd studenti

(kali㉿kali)-[~/studenti]
$ mkdir {nicola,matteo}

(kali㉿kali)-[~/studenti]
$ ls
anna matteo nicola
```

Figura 3

Sempre con il comando "mkdir" andiamo a creare le sottocartelle "amici" nella cartella "matteo" e dentro "nicola" andiamo a creare "lavoro" e "scuola"

```
(kali㉿kali)-[~/studenti]
$ tree
.
├── anna
│   └── casa
├── matteo
│   └── amici
├── nicola
│   ├── lavoro
│   └── scuola
└── 
```

Figura 4

Visto che, se immaginiamo lo schema iniziale in livelli, abbiamo finito l'ultimo livello delle cartelle e ci manca inserire soltanto i "documenti.txt" li creiamo con il comando "touch" e mostriamo così il risultato con "tree".

```
(kali㉿kali)-[~]
$ tree
.
├── Desktop
├── Documents
├── dos
├── Downloads
├── Music
├── Pictures
├── Public
├── ssl-test
│   ├── cert.pem
│   ├── index.html
│   └── key.pem
├── student
├── studenti
│   ├── anna
│   │   └── casa
│   ├── matteo
│   │   └── amici
│   └── nicola
│       ├── lavoro
│       │   └── compito.doc
│       └── scuola
│           └── relazione.doc
├── Templates
├── tmp
│   └── risultati.doc
├── Videos
└── windows
```

Figura 5

Nelle prossime due figure porto tutte le cartelle con i loro contenuti in una cartella principale nominata "HOME" e sposto "compito.txt" dalla cartella "lavoro" a "scuola". In questo modo (Figura 8) siamo giunti alla situazione di partenza mostrata nella figura 0

```
HOME
├── dos
├── studenti
│   ├── anna
│   │   └── casa
│   ├── matteo
│   │   └── amici
│   └── nicola
│       ├── lavoro
│       │   └── compito.doc
│       └── scuola
│           └── relazione.doc
├── tmp
│   └── risultati.doc
└── windows
```

Figura 6

```
(kali㉿kali)-[~/HOME/studenti/anna/casa]
$ mv ~/HOME/studenti/nicola/lavoro/compito.doc ~/HOME/studenti/nicola/scuola/
Completing external command
```

Figura 7

```
(kali㉿kali)-[~]
$ tree
.
├── Desktop
├── Documents
├── Downloads
├── HOME
│   ├── dos
│   ├── studenti
│   │   ├── anna
│   │   │   ├── casa
│   │   │   ├── matteo
│   │   │   │   ├── amici
│   │   │   │   ├── nicola
│   │   │   │   ├── lavoro
│   │   │   │   └── scuola
│   │   │       ├── compito.doc
│   │   │       └── relazione.doc
│   │   └── tmp
│   │       └── risultati.doc
│   └── windows
├── Music
├── Pictures
├── Public
├── ssl-test
│   ├── cert.pem
│   ├── index.html
│   └── key.pem
├── student
├── Templates
└── Videos

23 directories, 6 files
```

Figura 8

Da questo momento inizieremo a svolgere le varie consegne che l'esercitazione ci vuole fare svolgere.

-Punto A Copia il file compito.doc (dalla directory scuola) nella directory corrente (casa)

Tramite il comando "cd" ci spostiamo nella cartella "casa". Poi con "cp" andiamo a copiare il documento e infine tramite il comando "ls" andiamo a visualizzare che il file è presente nella cartella

```
(kali㉿kali)-[~]
$ cd HOME/studenti/anna/casa

(kali㉿kali)-[~/HOME/studenti/anna/casa]
$ cp ../../nicola/scuola/compito.doc .

(kali㉿kali)-[~/HOME/studenti/anna/casa]
$ ls
compito.doc
```

Figura 9

-Punto B Sposta il file relazione.doc nella directory corrente (casa)

Tramite il comando “mv” andiamo a spostare il documento in questione e lo visualizziamo con tree

```
(kali㉿kali)-[~/HOME/studenti/anna/casa]
$ mv ../../nicola/scuola/relazione.doc .

(kali㉿kali)-[~/HOME/studenti/anna/casa]
$ tree
.
├── compito.doc
└── relazione.doc
```

Figura 10

-Punto C Cancella la cartella \tmp

Con il tasto “rm” andiamo a rimuovere la cartella \tmp nel caso nostro aggiungiamo “-rf” per far eliminare oltre alla cartella anche il loro contenuto

```
(kali㉿kali)-[~/HOME]
$ rm -rf tmp

(kali㉿kali)-[~/HOME]
$ tree
.
├── dos
├── studenti
│   ├── anna
│   │   └── casa
│   │       ├── compito.doc
│   │       └── relazione.doc
│   ├── matteo
│   │   └── amici
│   └── nicola
│       ├── lavoro
│       └── scuola
│           └── compito.doc
└── windows

11 directories, 3 files
```

Figura 11

-Punto D Creare il file pippo.txt nella cartella lavoro

Tramite il comando “touch” andiamo a creare il file pippo.txt

```
(kali㉿kali)-[~/HOME]
$ tree
.
├── dos
├── studenti
│   ├── anna
│   │   └── casa
│   │       ├── compito.doc
│   │       └── relazione.doc
│   ├── matteo
│   │   └── amici
│   ├── nicola
│   │   ├── lavoro
│   │   │   ├── pippo.txt
│   │   └── scuola
│   │       └── compito.doc
└── windows
```

Figura 12

-Punto E Cambiare gli attributi del file pippo.txt e renderlo scrivibile e leggibile solo per il proprietario, mentre per tutti gli altri solo leggibile

Innanzitutto andiamo nella cartella lavoro tramite al comando “cd”, una volta dentro tramite il comando “ls -l” andiamo a vedere chi e quali privilegi hanno, poi tramite il comando “chmod” andiamo a dare i privilegi come richiesto dall’esercizio

```
(kali㉿kali)-[~/HOME]
$ cd studenti/nicola/lavoro

(kali㉿kali)-[~/HOME/studenti/nicola/lavoro]
$ ls -l pippo.txt
-rw-rw-r-- 1 kali kali 0 Jul 23 14:33 pippo.txt

(kali㉿kali)-[~/HOME/studenti/nicola/lavoro]
$ chmod g-w pippo.txt

(kali㉿kali)-[~/HOME/studenti/nicola/lavoro]
$ ls -l pippo.txt
-rw-r--r-- 1 kali kali 0 Jul 23 14:33 pippo.txt

(kali㉿kali)-[~/HOME/studenti/nicola/lavoro]
$
```

Figura 13

-Punto F Nascondere il contenuto della cartella Anna

In realtà tramite il comando “mv” in realtà rinominato la cartella “anna” in “.anna” che su Linux diventa un file nascosto grazie al punto iniziale.

```
(kali㉿kali)-[~/HOME/studenti]
$ mv anna/ .anna

(kali㉿kali)-[~/HOME/studenti]
$ ls
matteo nicola
```

Figura 14

-Punto G Spostarsi nella cartella lavoro e visualizzare il contenuto del file pippo.txt

Innanzitutto prima di proseguire con questo punto andiamo a scrivere dentro il file pippo.txt attraverso il comando “vim”. Dopo aver fatto questo usiamo il comando “cat” per andare a visualizzare appunto il contenuto del documento

```
(kali㉿kali)-[~/HOME]
$ cd studenti/nicola/lavoro

(kali㉿kali)-[~/HOME/studenti/nicola/lavoro]
$ cat pippo.txt
ciao Valerio
```

Figura 15

-Punto H Rimuovere la cartella amici

In questo caso usiamo il comando “rmdir” per andare ad eliminare la cartella “amici”

```
(kali㉿kali)-[~/HOME]
$ rmdir studenti/matteo/amici

(kali㉿kali)-[~/HOME]
$ tree
.
├── dos
├── studenti
│   ├── matteo
│   └── nicola
│       ├── lavoro
│       │   ├── pippo.txt
│       │   └── scuola
│       │       └── compito.doc
│       └── pippo.txt
└── windows
```

Figura 16

-Punto I Rimuovere tutte le cartelle precedentemente create

Andiamo ora ad eliminare il resto delle cartelle create in precedenza attraverso il comando “rm -rf” con il nome delle cartelle e ritorniamo così al punto di partenza.

```
(kali㉿kali)-[~/HOME]
$ tree
.
├── dos
├── studenti
│   ├── matteo
│   └── nicola
│       ├── lavoro
│       │   ├── pippo.txt
│       │   └── scuola
│       │       └── compito.doc
│       └── pippo.txt
└── windows

8 directories, 3 files

(kali㉿kali)-[~/HOME]
$ rm -rf dos studenti windows

(kali㉿kali)-[~/HOME]
$ ls

(kali㉿kali)-[~/HOME]
```

Figura 17

PARTE FACOLTATIVA

Facoltativo:

who	lista utenti collegati
who am i	chi sono io
jobs	elenco lavori sul terminale
&	apre processo in background
fg	metti in foreground
bg	metti in background
ps	elenco processi
kill	termina processo

Provare i comandi:

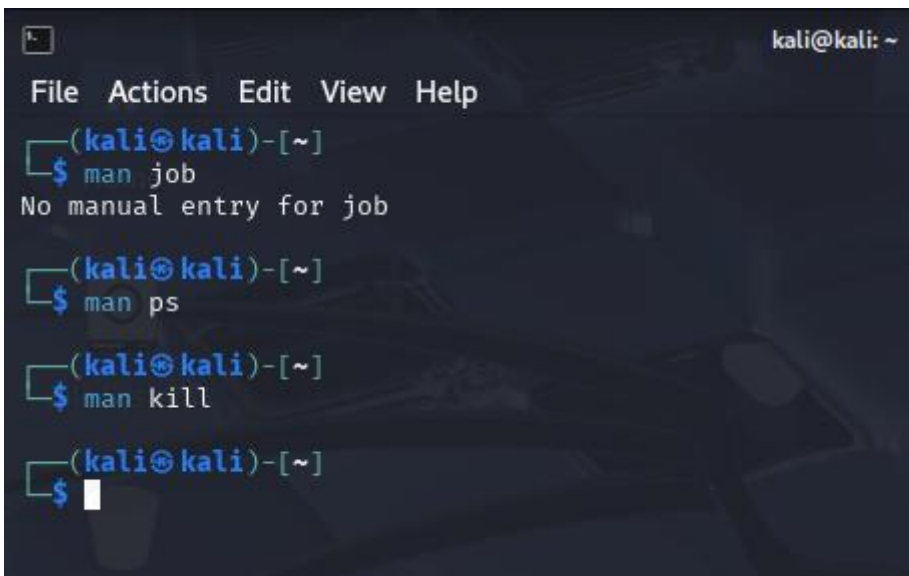
w
who
who am i

Esercizi - processi:

1. Aprire un terminale
2. leggere il manuale del comando job, ps e kill
3. lanciare il comando vi pippo
4. aprire un nuovo terminale e visualizzare tutti i propri processi...
5. cercare di terminare (killare) il processo vi per sbloccare il terminale precedente
6. lanciare il comando firefox in background
7. portarlo in background
8. cercare di terminare il processo firefox
9. verificare quanto spazio si sta occupando su disco

La parte facoltativa dell'esercitazione punta a farci familiarizzare con i comandi di base per monitorare e controllare i processi e a spostare processi da foreground e background e l'uso di alcune delle più comuni applicazioni.

Come prima parte apriamo un terminale e andiamo a vedere i manuali degli applicativi job, ps e kill tramite "man" i primi due servono per vedere l'elenco rispettivamente dei lavori sul terminale e il secondo per i processi infine l'applicativo "kill" terminano un processo.



```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ man job  
No manual entry for job  
(kali@kali)-[~]  
$ man ps  
(kali@kali)-[~]  
$ man kill  
(kali@kali)-[~]  
$
```

Sotto vediamo l'esempio del manuale "kill" aperto nel quale possiamo vedere tutte le funzioni dell'applicazione e come attivarle.


```
kali-linux-2025.1c-virtualbox-amd64 [In esecuzione] - Oracle VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto
1 2 3 4
kali@kali: ~
File Actions Edit View Help
KILL(1) User Commands KILL(1)
NAME
    kill - send a signal to a process
SYNOPSIS
    kill [options] <pid> [...]
DESCRIPTION
    The default signal for kill is TERM. Use -l or -L to list available signals. Particularly useful signals include HUP, INT, KILL, STOP, CONT, and 0. Alternate signals may be specified in three ways: -9, -SIGKILL or -KILL. Negative PID values may be used to choose whole process groups; see the PGID column in ps command output. A PID of -1 is special; it indicates all processes except the kill process itself and init.
OPTIONS
    <pid> [...]
        Send signal to every <pid> listed.

    -<signal>
    -s <signal>
    --signal <signal>
        Specify the signal to be sent. The signal can be specified by using name or number. The behavior of signals is explained in signal(7) manual page.

    -q, --queue value
        Use sigqueue(3) rather than kill(2) and the value argument is used to specify an integer to be sent with the signal. If the receiving process has installed a handler for this signal using the SA_SIGINFO flag to sigaction(2), then it can obtain this data via the si_value field of the siginfo_t structure.

    -l, --list [signal]
        List signal names. This option has optional argument, which will convert signal number to signal name, or other way round.

    -L, --table
        List signal names in a nice table.

NOTES
    Your shell (command line interpreter) may have a built-in kill command. You may need to run the command described here as /bin/kill to solve the conflict.

EXAMPLES
    kill -9 -1
        Kill all processes you can kill.

    kill -l 11
        Translate number 11 into a signal name.

    kill -L
        List the available signal choices in a nice table.
Manual page kill(1) line 1 (press h for help or q to quit)
```

Dopo apriamo il terminale lanciando il comando “vi pippo” e tenendolo attivo apriamo un altro terminale avviando il comando aux

```
kali-linux-2025.1c-virtualbox-amd64 [In esecuzione] - Oracle VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto
1 2 3 4
kali@kali: ~
File Actions Edit View Help
kali 1132 0.7 2.3 621988 46672 ? Sl 04:55 0:32 nm-applet
kali 1134 0.0 1.1 336400 23780 ? Sl 04:55 0:00 xfce4-power-manager
kali 1135 0.0 0.0 12436 2000 ? Ssl 04:55 0:00 xcage -e Super_L Control_L Escape
kali 1145 0.0 0.8 190924 16236 ? Sl 04:55 0:00 /usr/libexec/polkit-mate-authenticatio
kali 1166 0.0 0.3 308448 6708 ? Sl 04:55 0:00 /usr/libexec/geoclue-2.0/demos/agent
kali 1169 0.0 2.0 64140 40464 ? S 04:55 0:00 /usr/bin/python3 /usr/share/system-con
kali 1194 0.0 0.4 857004 8880 ? Sl 04:55 0:00 xiccd
colord 1201 0.0 0.7 315836 14228 ? Ssl 04:55 0:00 /usr/libexec/colord
kali 1221 0.0 0.6 426228 13024 ? Ssl 04:55 0:00 /usr/libexec/gvfs-udisks2-volume-monit
root 1232 0.0 0.6 404016 13652 ? Ssl 04:55 0:00 /usr/libexec/udisks2/udisksd
kali 1258 0.0 0.3 308796 6832 ? Ssl 04:55 0:00 /usr/libexec/gvfs-gphoto2-volume-monit
kali 1263 0.0 0.4 389320 8380 ? Ssl 04:55 0:00 /usr/libexec/gvfs-afc-volume-monitor
kali 1280 0.0 0.3 307844 6556 ? Ssl 04:55 0:00 /usr/libexec/gvfs-mtp-volume-monitor
kali 1287 0.0 0.3 307748 6388 ? Ssl 04:55 0:00 /usr/libexec/gvfs-goa-volume-monitor
kali 1296 0.0 0.8 377496 18084 ? Ssl 04:55 0:00 /usr/libexec/bluetooth/obexd
kali 1297 0.0 0.3 168792 6616 ? Ssl 04:55 0:00 /usr/libexec/gvfsd-metadata
kali 1301 0.0 0.4 534004 8904 ? Sl 04:55 0:00 /usr/libexec/gvfsd-trash --spawner :1.
root 1334 0.0 0.0 0 0 ? I< 04:55 0:00 [kworker/u12:2-ttm]
kali 1418 0.0 0.9 482052 18680 ? Ssl 04:55 0:00 /usr/libexec/xdg-desktop-portal
kali 1427 0.0 0.3 308660 6384 ? Ssl 04:55 0:00 /usr/libexec/xdg-permission-store
kali 1432 0.0 0.3 532448 6896 ? Ssl 04:55 0:00 /usr/libexec/xdg-document-portal
root 1438 0.0 0.0 2576 1892 ? Ss 04:55 0:00 fusermount3 -o rw,nosuid,nodev,fsname=
kali 1442 0.0 0.9 406448 18852 ? Ssl 04:55 0:00 /usr/libexec/xdg-desktop-portal-gtk
root 3725 0.0 0.0 0 0 ? I 05:00 0:00 [kworker/u9:4-events_unbound]
root 9044 0.0 0.0 0 0 ? I 05:11 0:00 [kworker/u9:1-flush-8:0]
root 13484 0.0 0.0 0 0 ? I< 05:20 0:00 [kworker/1:1H-kblockd]
root 17355 0.0 0.0 0 0 ? I< 05:28 0:00 [kworker/1:2H]
root 20340 0.0 0.0 0 0 ? I 05:34 0:00 [kworker/0:1-events]
root 24987 0.0 0.0 0 0 ? I< 05:43 0:00 [kworker/u13:1-ttm]
root 25236 0.0 0.0 0 0 ? I< 05:44 0:00 [kworker/u12:0-ttm]
root 26448 0.0 0.0 0 0 ? I 05:46 0:00 [kworker/u10:3-events_unbound]
root 29378 0.0 0.0 0 0 ? I 05:52 0:00 [kworker/u10:0-flush-8:0]
root 32141 0.0 0.0 0 0 ? I 05:58 0:00 [kworker/1:2-ata_sff]
root 33242 0.0 0.0 0 0 ? I 06:00 0:00 [kworker/u10:2-events_unbound]
root 34628 0.0 0.0 0 0 ? I 06:03 0:00 [kworker/1:1-events]
kali 36423 0.0 0.3 307528 6304 ? Ssl 06:07 0:00 /usr/lib/x86_64-linux-gnu/xfce4/xfconf
kali 36715 0.1 3.1 578316 63340 ? Sl 06:07 0:00 /usr/bin/qterminal
kali 36721 0.0 0.3 10260 6392 pts/0 Ss 06:07 0:00 /usr/bin/zsh
kali 36913 0.0 0.5 16124 10752 pts/0 Sl+ 06:08 0:00 vi pippo
kali 36966 0.3 3.1 578320 63228 ? Sl 06:08 0:00 /usr/bin/qterminal
kali 36972 0.1 0.3 10264 6400 pts/1 Ss 06:08 0:00 /usr/bin/zsh
root 37213 0.0 0.0 0 0 ? I 06:08 0:00 [kworker/1:0-events_freezable_pwr_effi
kali 37264 0.0 0.2 9484 4148 pts/1 R+ 06:08 0:00 ps aux

(kali@kali)-[~]
$ ps aux | grep pippo
kali 36913 0.0 0.5 16124 10880 pts/0 Sl+ 06:08 0:00 vi pippo
kali 41524 0.0 0.1 6520 2216 pts/1 S+ 06:17 0:00 grep --color=auto pippo

(kali@kali)-[~]
$
```

Poi andiamo a chiudere l'applicazione aperte tramite il comando "kill" aggiungendo il numero associato in questo caso 36913 come fatto nello screen seguente

```
(kali㉿kali)-[~]
$ ps aux | grep pippo
kali      36913  0.0  0.5 16124 10880 pts/0    Sl+  06:08   0:00 vi pippo
kali      41524  0.0  0.1  6520  2216 pts/1    S+   06:17   0:00 grep --color=auto pippo

(kali㉿kali)-[~]
$ kill -9 36913

(kali㉿kali)-[~]
$
```

Dopo aver effettuato questo andiamo a vedere come nella pagina del terminale con l'applicativo aperto ci prospetta la dicitura killed

```
zsh: killed 114 vi pippo
~ 11 1166 0.0 0.3 308448 6708 ?
kali 1169 0.0 2.0 64140 40484 ?
(kali㉿kali)-[~] 0 0.4 857004 8880 ?
$ 1201 0.0 0.7 315836 14228 ?
kali 1221 0.0 0.6 425228 19024 ?
root 1232 0.0 0.6 404015 19552 ?
kali 1258 0.0 0.3 308796 6832 ?
```

Come penultima parte dell'esercizio ci chiede di aprire firefox, poi tramite terminale prima sospenderlo e lo facciamo tramite la combinazione tasti CTRL+Z, poi metterla in background tramite il comando "bg"

```
kali@kali: ~
File Actions Edit View Help
(kali㉿kali)-[~]
$ firefox
^Z
zsh: suspended firefox
(kali㉿kali)-[~]
$ bg
[1] + continued firefox
(kali㉿kali)-[~]
$
```

Infine mostriamo come chiuderlo tramite il comando kill in questo caso aggiungiamo il "%" con il numero che corrisponde all'applicativo che vogliamo chiudere "1"

```
(kali㉿kali)-[~]
$ kill %1

(kali㉿kali)-[~]
$
[1] + terminated firefox
(kali㉿kali)-[~]
$
```