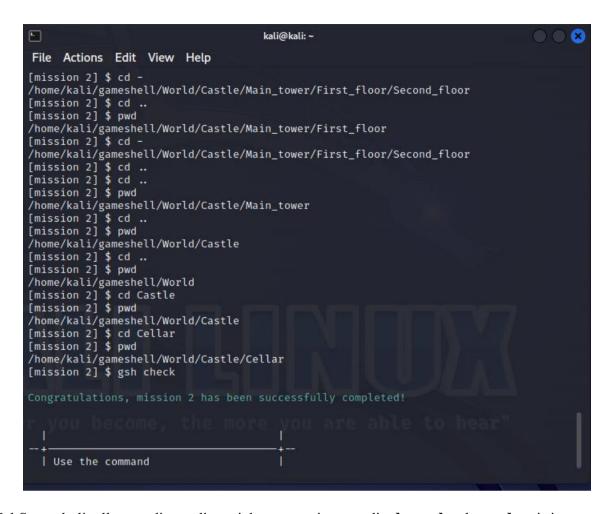
```
kali@kali: ~
File Actions Edit View Help
[mission 1] $ cd castle
bash: cd: castle: No such file or directory
[mission 1] $ cd Castle
[mission 1] $ pwd
/home/kali/gameshell/World/Castle
[mission 1] $ ls
Cellar Great_hall Main_building Main_tower Observatory
[mission 1] $ cd Main_tower
[mission 1] $ pwd
/home/kali/gameshell/World/Castle/Main_tower
[mission 1] $ ls
First_floor
[mission 1] $ cd First_floor
[mission 1] $ pwd
/home/kali/gameshell/World/Castle/Main_tower/First_floor
[mission 1] $ ls
Second_floor
[mission 1] $ cd Second_floor
[mission 1] $ ls
Top_of_the_tower
[mission 1] $ cd Top_of_the_tower
[mission 1] $ ls
[mission 1] $ pwd
/home/kali/gameshell/World/Castle/Main_tower/First_floor/Second_floor/Top_of_the_tow
[mission 1] $ gsh check
```

In questo primo livello vediamo l'utilizzo di **cd** per spostarci di posizione, **pwd** per ottenere la nostra posizione attuale, **ls** per sapere verso quali directory potersi spostare.

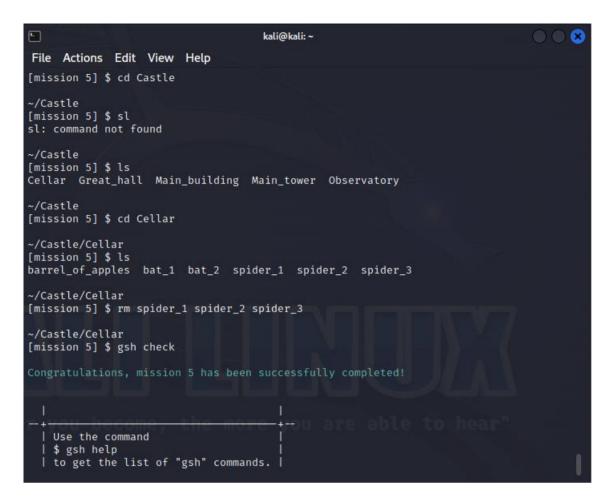


Nel Secondo livello prendiamo dimestichezza con i comandi  $\mathbf{cd}$  .. e  $\mathbf{cd}$  -, dove  $\mathbf{cd}$  .. ti riporta sempre alla cartella precedente rispetto alla directory in cui ci si trova, mentre  $\mathbf{cd}$  - riporta una singola volta all'interno della directory precedente.



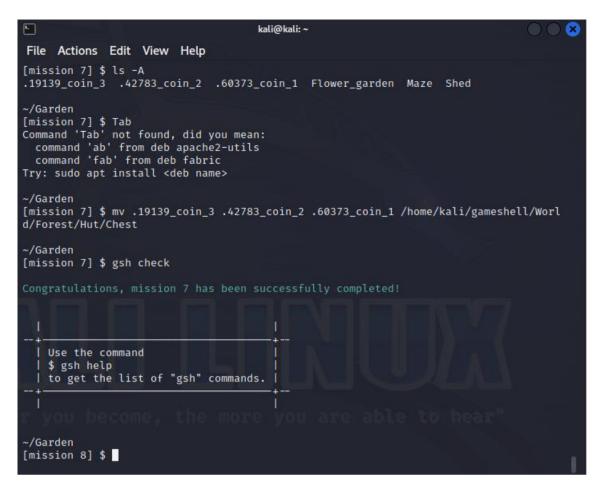
All'interno del quarto livello abbiamo come obbiettivo la creazione di due directory, con nome **Hut** e **Chest** all'interno dei posti decisi dal gioco, ovvero nella foresta.

Questa operazione la si può ottenere tramite il comando **mkdir**, in questo caso il comando viene inserito già all'interno della cartella **Forest**.

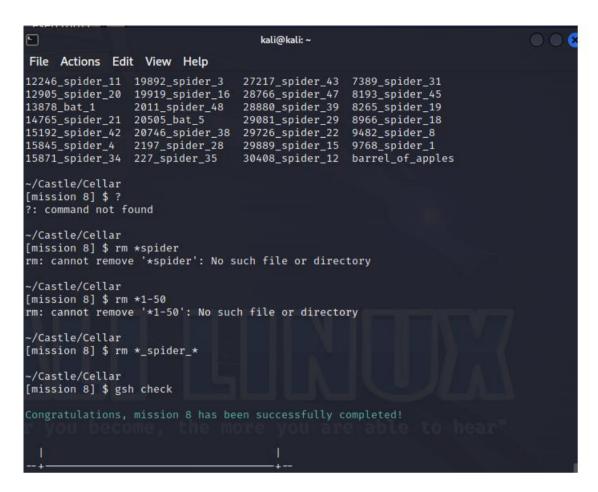


All'interno del quinto livello l'obbiettivo è quello di eliminare tutti i ragni. Per fare ciò,utiliziamo ls per scoprire il nominativo dei file e utiliziamo il comando **rm** seguito dal nominativo del file da eliminare, in questo caso: **spider\_1**, **spider\_2** e **spider\_3**.

All'interno del sesto livello facciamo uso del comando **mv** -oltre ai precedenti- per poter spostare i file **coin\_1** , **coin\_2** e **coin\_3** all'interno della directory della **Chest**, ovvero: /home/kali/gameshell/world/Forest/Hut/Chest



Nel Settimo livello utiliziamo tutti i precendi comandi e ne aggiungiamo uno nuovo, ovvero **ls -A** che ci permetterà di controllare anche le directory non visibili con il comando **ls**.



Nell'ottava missione utiliziamo il comando **rm** per eliminare le directory **spider**, seguito da \* \_ **spider** \_ \* in maniera tale da non dover scrivere tutti i numeri precedenti e successivi al \_ **spider** \_

```
kali@kali: ~
File Actions Edit View Help
.1938_spider_20 .24705_spider_12 .32482_spider_41
[mission 9] $ rm
rm: missing operand
Try 'rm --help' for more information.
~/Castle/Cellar
[mission 9] $ rm *.*_spider_*
rm: cannot remove '*.*_spider_*': No such file or directory
~/Castle/Cellar
[mission 9] $ rm *._spider_*
rm: cannot remove '*._spider_*': No such file or directory
~/Castle/Cellar
[mission 9] $ rm .*_spider_*
~/Castle/Cellar
[mission 9] $ gsh check
                                       Congratulations !
          _.-:::::' ||
|--:::::' ||
                                       From now on, the ``ls`` command will automatically
                                       show a "/" character at the end of directories.
```

Nella missione 9 ricerchiamo i file nascosti tramite **ls -A**.

Fatto ciò, procediamo con l'inserimento anche del . (che indica il file nascosto) all'interno delle \*.

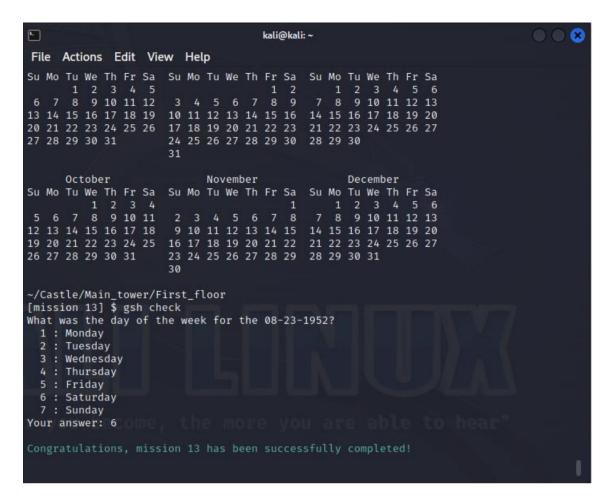
Ciò serve al non dover scrivere i numeri precedenti e successivi al file \_ spider\_

```
kali@kali: ~
File Actions Edit View Help
~/Castle/Cellar
[mission 10] $ cd
[mission 10] $ ls
Castle/ Forest/ Garden/ Mountain/ Stall/
[mission 10] $ cd Castle
~/Castle
[mission 10] $ ls
Cellar/ Great_hall/ Main_building/ Main_tower/ Observatory/
~/Castle
[mission 10] $ cd Great_hall
~/Castle/Great_hall
~/Castle/Great_Mate
[mission 10] $ ls
32936_suit_of_armour 6537_decorative_shield standard_2 standard_4
62274 stag_head standard_1 standard_3
~/Castle/Great_hall
[mission 10] $\(^\text{cp}\) standard_1 standard_2 standard_3 standard_4 /home/kali/gameshell/W
orld/Forest/Hut/Chest
~/Castle/Great_hall
[mission 10] $ gsh check
```

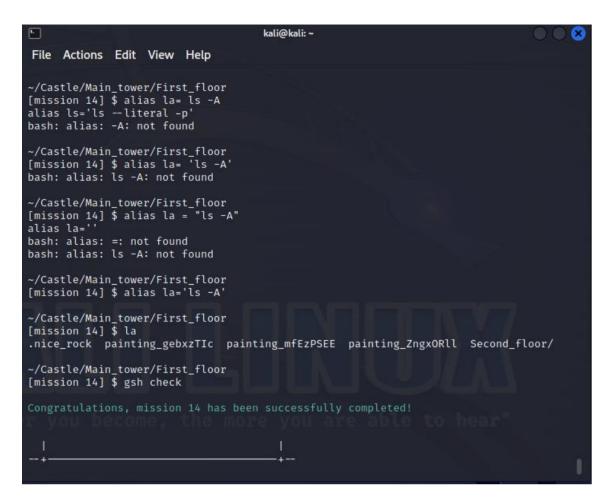
Nella missione 10 utiliziamo il comando **cp** per creare una copia dei file all'interno della directory **Chest** precedentemente creata.



Nella missione 12 utiliziamo il comando **ls -l** per ricercare i file in ordine, dal più vecchio al più recente, per poi copiare il più vecchio tramite il comando **cp** nella directory **Chest**.

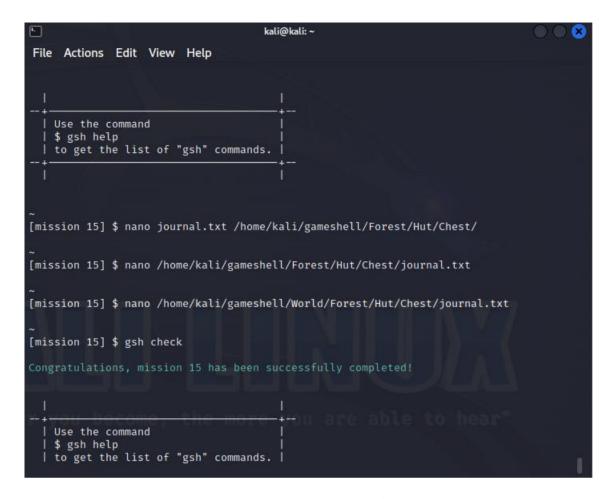


Nella missione 13 utilizziamo il comando **cal** per sapere il mese del corrente anno, mentre **cal 1952** (in questo caso è stato utilizzato 1952, ma potrebbe essere usato un anno qualsiasi) per sapere il calendario dell'anno in questione, per poter risolvere l'enigma del giorno.



Nella missione 14 avremo da creare un alias per creare una scorciatoia. Ciò è possibile utilizzando il comando **alias** (qualcosa, in questo caso la) **la** = '**ls** -A', creando una scorciatoia per trovare i file nascosti.

In questo modo potrà essere utilizzato semplicemente la anzichè ls -A.



Nella missione 15 avremo da creare un file di testo dal nome **journal.txt** e da inserire all'interno della directory **Chest** creata a inizio gioco. Ciò è fattibile tramite **nano** seguito dal comando della directory finale con il nome del file da creare, ovvero: **journal.txt.** 

```
kali@kali: ~
File Actions Edit View Help
bash: alias: nano: not found
bash: alias: /home/kali/gameshell/World/Forest/Hut/Chest/journal.txt: not found
bash: alias: =: not found
bash: alias: nano journal: not found
[mission 16] $ alias nano /home/kali/gameshell/World/Forest/Hut/Chest/journal.txt =
'nano journal.txt'
bash: alias: nano: not found
bash: alias: /home/kali/gameshell/World/Forest/Hut/Chest/journal.txt: not found
bash: alias: =: not found
bash: alias: nano journal.txt: not found
[mission 16] $ alias journal = 'nano /home/kali/gameshell/World/Forest/Hut/Chest/jou
rnal.txt
bash: alias: journal: not found
bash: alias: =: not found
bash: alias: nano /home/kali/gameshell/World/Forest/Hut/Chest/journal.txt: not found
[mission 16] $ alias journal='nano /home/kali/gameshell/World/Forest/Hut/Chest/journ
al.txt'
[mission 16] $ gsh check
```

Nella missione 16 avremo da creare un alias per il recupero del file journal appena creato. Per far ciò abbiamo bisogno di scrivere alias journal = 'nano /home/kali/gameshell/World/Forest/Hut/Chest/journal.txt'



In questa missione alla ricerca dei file nascosti e a tempo, si ha un forte utilizzo del tasto **Tab** per accelerare la scrittura della lunga stringa di lettere e numeri.

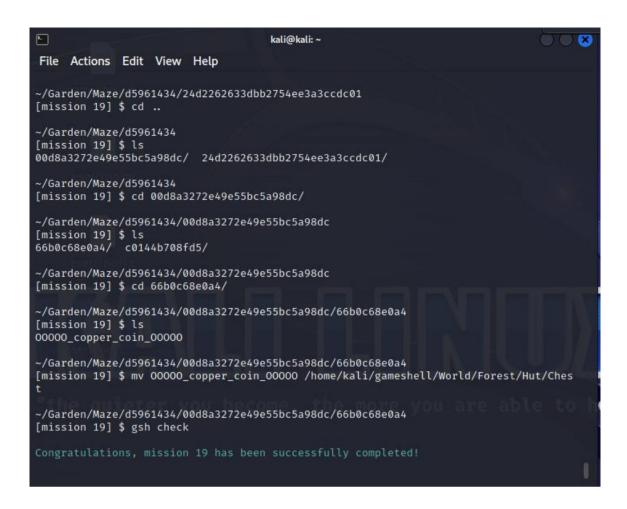
```
kali@kali: ~
File Actions Edit View Help
  | to get the list of "gsh" commands. |
~/Castle
[mission 18] $ xeyes &
[1] 92586
~/Castle
[mission 18] $ xeyes
^C
~/Castle
[mission 18] $ gsh check
   Use the command
    $ gsh help
  I to get the list of "gsh" commands.
~/Castle
[mission 19] $ kill 92586
[1]+ Terminated
                               xeyes
```

In questa missione avremo da inizializzare il programma **xeyes** che farà si che si apra un programma con degli occhi che seguono i movimenti del mouse.

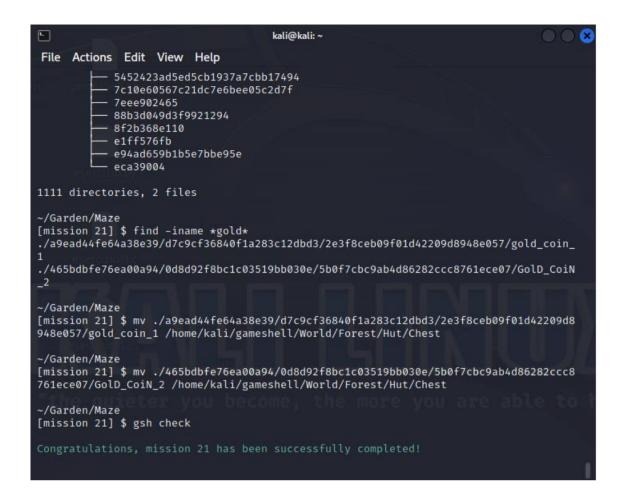
Dobbiamo bloccare il programma tramite l'utilizzo di ctrl C.

Dobbiamo anche inizializzare il programma in background tramite il comando xeyes &.

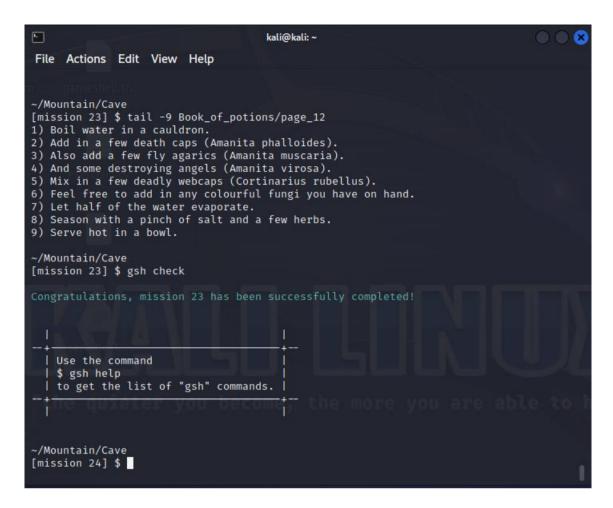
Per chiuderlo dobbiamo utilizzare il comando **kill** seguito dal numero del processo, in questo caso **92586**, ovvero: **kill 92586** 



In questa missione possiamo utilizzare una ricerca manuale della moneta di rame tramite i comandi **ls** e **cd** o una ricerca delle diverse cartelle -e file annessi- con il comando **tree**. Poi ci approcciamo allo spostamento della moneta nella **Chest** creata a inizio gioco.



In questo livello ricerchiamo il nome del file interessato tramite il comando find -iname \* gold \*



In questa missione utilizzeremo i comandi **cat**, **tail FILE** e **tail -n**.

Con il comando **cat** apriamo il file vedendone il contenuto.

Con il comando **tail** faremo una stampa delle ultime 10 linee con il comando **tail -n** faremo la stampa del numero di linee inserite dall'utente.