

# PROGETTO FINALE M2

Mungiovì Fabio



## TASK

---

Il seguente report vuole essere una guida del gioco GameShell.

Questo semplice gioco, eseguibile dal terminale di Linux, propone delle “missioni” da svolgere che, per essere portate a termine, richiedono l'utilizzo dei comandi di base del terminale, permettendo in questo modo di familiarizzare con essi.

La struttura del gioco non è altro che un sistema di cartelle, sotto cartelle e file, con nomi che richiamano un'ambientazione “medioevale fantasy”, e ad ogni livello del gioco viene insegnato un diverso comando del terminale, per muoversi tra cartelle, creare/eliminare file ecc...

```
(\
Welcome to GameShell!

Using and administering a computer running a Unix-like
operating system (e.g., Linux, BSD or macOS) requires
using a command line interface called "shell". In this
game you will learn how to use the shell by journeying
through various missions that will teach you to perform
standard tasks such as:
- creating directories,
- creating, viewing and editing files,
- searching for files satisfying certain criteria,
- running and interrupting programs,
- etc.

During your adventure you will need to visit locations,
interact with people, and find various objects. However,
you have to keep in mind that these are standard
directories, programs, and files (often with random
contents). So, for example:
- "build a hut" means creating a directory called
"Hut", and
- "put a coin in your chest" means moving a file
called "coin" to a directory called "Chest".

(/*
[Press space to continue, q to quit.]
```

Schermata di presentazione di GameShell

I due comandi essenziali per questo gioco sono:

**gsh goal** : Restituisce a schermo l'obiettivo della missione

**gsh check** : Controlla se l'obiettivo della missione è stato raggiunto

Fatte queste dovute presentazioni, possiamo iniziare l'avventura...

# # MISSION 1

```
[mission 1] $ gsh goal
```

## 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.

## Remarks

UPPERCASE words appearing in commands are meta-variables: you need to replace them by appropriate (string) values.

Most filesystems treat uppercase and lowercase characters differently. Make sure you use the correct path.

La prima missione consiste nel raggiungere la cima della torre del castello, o meglio la cartella *Top\_of\_the\_tower*.

Prendiamo quindi familiarità con i comandi base per spostarsi nelle cartelle da terminale che sono:

**pwd** Visualizza la directory attuale

**ls** Visualizza il contenuto della directory dove ci troviamo

**cd** Cambia la posizione nella directory specificata

Come possiamo notare dalle istruzioni restituite dal comando `gsh goal`, ad ogni livello vengono indicati i comandi utili al fine del completamento dello stesso.

Di seguito l'esecuzione fino al raggiungimento dell'obiettivo.

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

```
[mission 1] $ pwd
/home/kali/gameshell/World
```

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

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

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

```
[mission 1] $ cd Castle
```

```
[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] $ gsh check
```

```
Congratulations, mission 1 has been successfully completed!
```

## # MISSION 2

---

```
[mission 2] $ gsh goal

()=(
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.
)()=
```

In questo livello esploriamo i seguenti comandi:

`cd ..` Sposta alla cartella superiore

`cd -` Torna all'ultima posizione

Li utilizziamo per tornare indietro fino all'ingresso del castello per dirigerci poi nelle cantine.

### Esecuzione

```
[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] $ pwd
/home/kali/gameshell/World/Castle/Main_tower/First_floor/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] $ cd ..

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

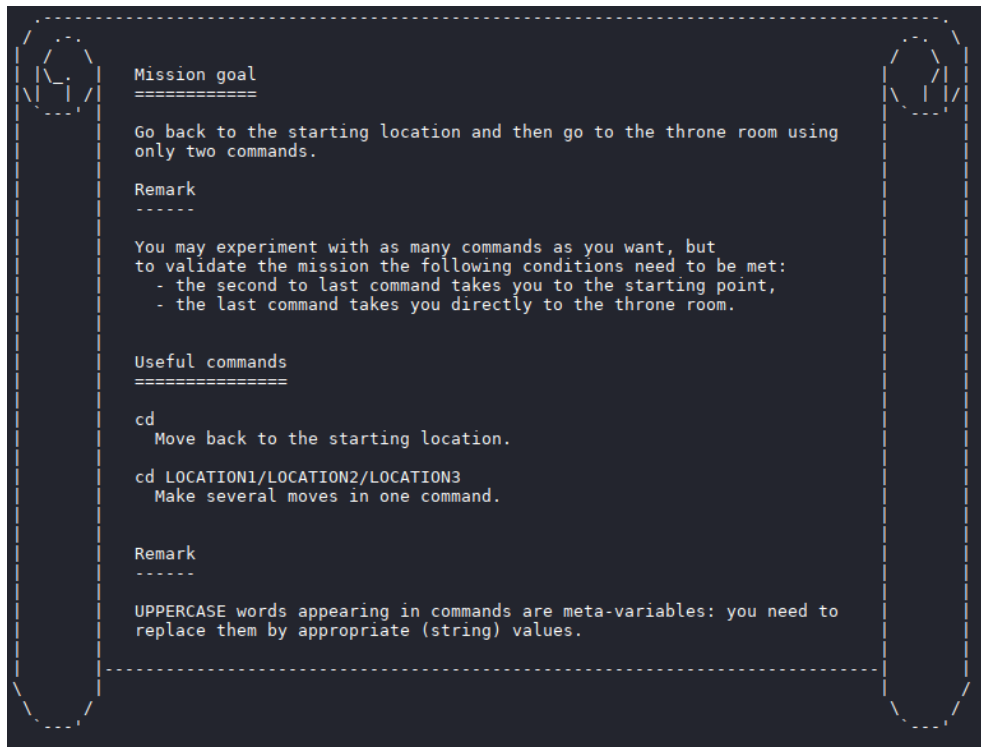
[use 'gsh help' to get a list of available commands]
[mission 2] $ ls
Cellar Great_hall Main_building Main_tower Observatory

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

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

Congratulations, mission 2 has been successfully completed!
```

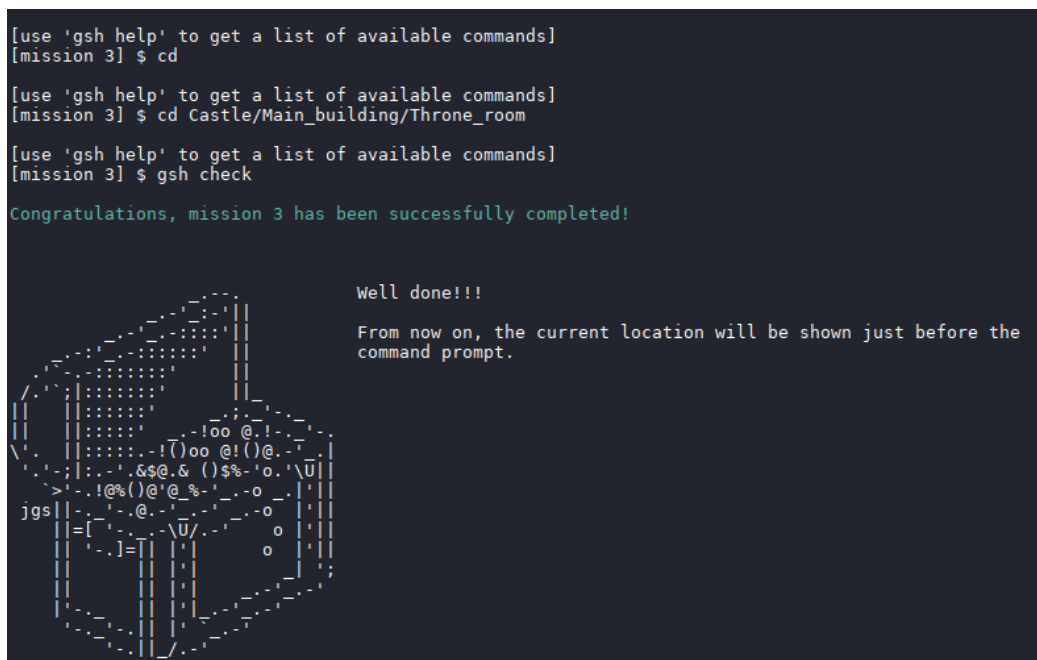
## # MISSION 3



Questa missione ci chiede di dirigerci in una certa posizione utilizzando solo 2 comandi, ci affideremo quindi al comando `cd` per tornare nella posizione di origine, e di seguito sempre al comando `cd`, seguito stavolta dal path di cartelle, fino alla destinazione richiesta.

Dopo l'esplorazione del mondo di gioco alla ricerca della *Throne\_room*, possiamo eseguire il comando dalla postazione di partenza:

```
cd Castle/Main_building/Throne_room
```



Abbiamo ricevuto anche un utile upgrade: Da ora la nostra posizione sarà sempre visibile, evitandoci l'uso del comando `pwd`

## # MISSION 4

---

```
()==(
| 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".
| )
) == (@==(
| .....
| )
```

Ci viene richiesto di creare un *Capanno* con uno *Scrigno* al suo interno dentro la *Foresta*.

Utilizziamo il comando `mkdir` per creare le cartelle *Hut* e *Chest* all'interno della directory richiesta.

### Esecuzione

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

~
[mission 4] $ cd Forest

~/Forest
[mission 4] $ mkdir Hut

~/Forest
[mission 4] $ cd Hut

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

~/Forest/Hut
[mission 4] $ gsh check

Congratulations, mission 4 has been successfully completed!
```

## # MISSION 5

```

^
┌-----┐
| /      | 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".
| /      |
└-----┐
^

```

Dobbiamo liberarci dei ragni in cantina (non dei pipistrelli, si dice portino fortuna), introduciamo quindi il comando `rm`, che rimuove in modo permanente i file.

Può essere utilizzato per rimuovere più file contemporaneamente, come nel nostro caso dove scriveremo:

```
rm spider_1 spider_2 spider_3
```

## Eliminando quindi 3 file con un solo comando

## Esecuzione

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

~
[mission 5] $ cd Castle/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!
```

## # MISSION 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.
/)
(*)
\\
))
^
```

Dobbiamo recuperare tutte le monete nel giardino e metterle nel nostro scrigno creato in precedenza.

Introduciamo il comando `mv` che, seguito dal nome del/dei file e dalla directory di destinazione, permette di spostare file.

```
mv coin_1 coin_2 coin_3 /home/kali/gameshell.1/World/Forest/Hut/Chest
```

### Esecuzione

```
~
[mission 6] $ cd

~
[mission 6] $ cd Garden

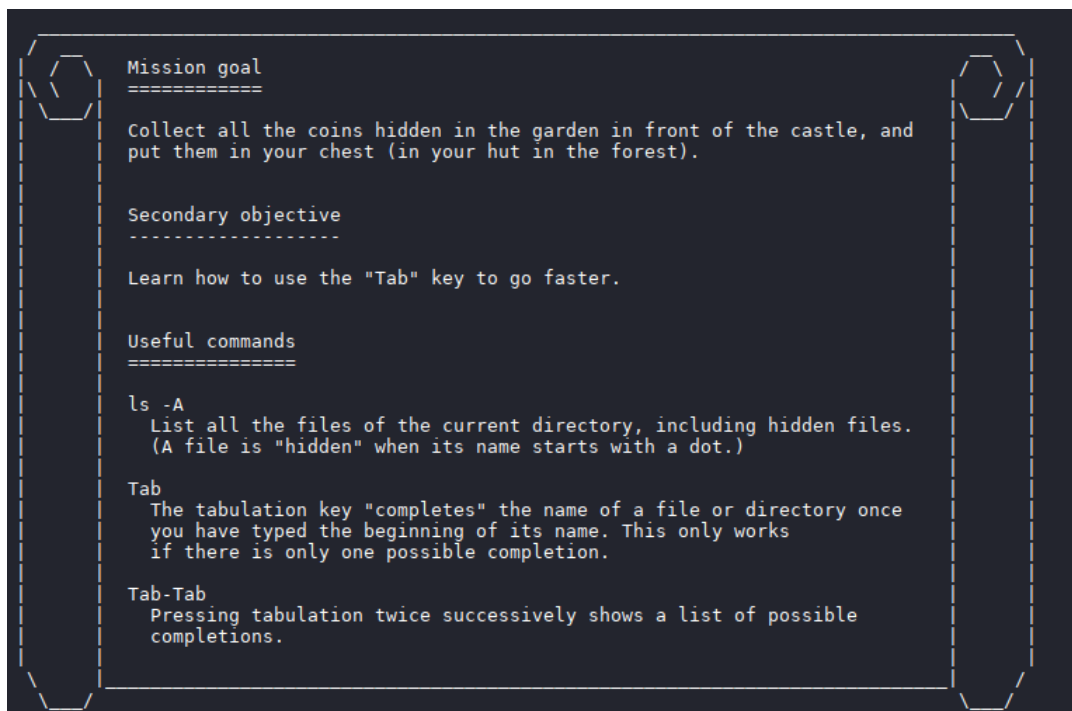
~/Garden
[mission 6] $ ls
coin_1 coin_2 coin_3 Flower_garden Maze Shed

~/Garden
[mission 6] $ mv coin_1 coin_2 coin_3 /home/kali/gameshell.1/World/Forest/Hut/Chest

~/Garden
[mission 6] $ gsh check

Congratulations, mission 6 has been successfully completed!
```

## # MISSION 7



In questa missione dovremmo trovare delle monete (file) nascosti, che posso essere visualizzati con il comando `ls -A`

I file nascosti vengono visualizzati con un "punto" davanti al nome: `.719_coin_1`

Ci viene anche consigliato l'utilizzo del tasto TAB per velocizzare la scrittura dei nomi dei file e delle directory

Una volta trovate anche le monete nascoste, spostiamo pure queste nel nostro scrigno

### Esecuzione

```
~/Garden
[mission 7] $ ls -A
.44890_coin_3 .51455_coin_2 .719_coin_1 Flower_garden Maze Shed

~/Garden
[mission 7] $ mv .44890_coin_3 .51455_coin_2 .719_coin_1 /home/kali/gameshell.1/World/Forest/Hut/Chest

~/Garden
[mission 7] $ gsh check

Congratulations, mission 7 has been successfully completed!
```



## # MISSION 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
  x*     --> error, no matching file
  *-?    --> file-1 Folder-1
  *-??   --> file-14

```

In questo livello ci vengono proposti alcuni patterns che permettono di selezionare più file senza necessariamente scriverli per intero.

I comandi `*` e `?` utilizzati come nella spiegazione di GameShell, premettono di facilitare le operazioni di selezione

L'obiettivo è ancora quello di eliminare i ragni dalla cantina, ma questa volta proveremo con i nuovi comandi proposti

Eseguendo quindi il comando `rm *spider*` verranno eliminati tutti i file contenenti la parola "spider"

### Esecuzione

```

~
[mission 8] $ cd Castle/Cellar

~/Castle/Cellar
[mission 8] $ ls
10401_spider_37 1799_spider_16 23283_spider_22 28618_spider_46 4245_spider_48 7994_spider_4
10789_spider_26 18147_spider_7 2337_spider_35 29123_spider_39 4625_spider_10 8163_spider_14
13942_spider_28 18269_spider_41 23380_bat_5 29877_spider_11 4803_spider_19 8354_spider_49
14104_spider_38 18445_spider_45 23601_spider_50 30008_spider_27 5285_bat_3 8635_spider_15
14676_spider_1 18551_spider_21 24725_spider_32 30143_spider_29 6565_bat_2 9945_spider_42
15086_spider_40 18555_spider_17 25057_spider_9 30644_spider_8 7127_spider_18 barrel_of_apples
16749_spider_12 19544_spider_6 25299_spider_20 30847_spider_3 7209_spider_25
17383_spider_2 20773_spider_13 25590_spider_31 31223_bat_4 7263_bat_1
17435_spider_33 22445_spider_30 25658_spider_34 31340_spider_24 7323_spider_36
17903_spider_5 22894_spider_43 26319_spider_47 32165_spider_23 7561_spider_44

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

~/Castle/Cellar
[mission 8] $ ls
23380_bat_5 31223_bat_4 5285_bat_3 6565_bat_2 7263_bat_1 barrel_of_apples

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

Congratulations, mission 8 has been successfully completed!

```

## # MISSION 9

```

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.

```

Questa missione è molto simile alla precedente, con la differenza che i file/ragni da eliminare ora sono nascosti.

Il comando per rimuoverli tutti insieme è: `rm *.spider*`

Per i file nascosti, prima del pattern di selezione va aggiunto un punto

## Esecuzione

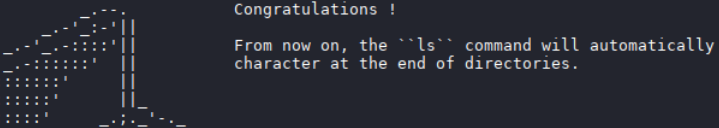
```
~/Castle/Cellar
[mission 9] $ ls -A
.10161_spider_27      .17516_spider_12      .24636_spider_6      .28803_bat_1          .7240_spider_18
.11088_spider_26     .18201_spider_44      .24861_spider_46     .29041_spider_15     7263_bat_1
.11675_spider_8      .18622_spider_24      .25032_spider_31     .30837_spider_48     .7731_spider_23
.12300_spider_38     .1962_spider_17       .25309_bat_2         31223_bat_4          .7852_spider_25
.12321_spider_49     .21193_spider_47      .25487_spider_37     .4095_spider_43      .83_spider_41
.12433_spider_45     .21273_spider_14      .2561_spider_28      .42_spider_32        .8867_spider_10
.12892_spider_9      .21613_bat_4          .25896_spider_11     .438_spider_33       .8871_spider_20
.15307_spider_3      .21712_spider_30      .26168_spider_34     5285_bat_3           .9507_spider_7
.15650_spider_50     .22249_bat_3          .26864_spider_42     .5953_spider_39      barrel_of_apples
.16279_spider_1      .22811_spider_2       .27048_spider_16     .6058_spider_21
.16474_bat_5          .2303_spider_13       .27402_spider_19     .6106_spider_29
.17444_spider_35     23380_bat_5           .27443_spider_22     6565_bat_2
.17507_spider_5      .23803_spider_36      .28046_spider_40     .7045_spider_4

~/Castle/Cellar
[mission 9] $ rm *.spider*

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

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

Abbiamo raggiunto anche un altro utile upgrade: Da ora le directory visualizzate con il comando `ls` verranno segnalate con un `/` in fondo



```

Congratulations !

From now on, the `ls` command will automatically show a "/"
character at the end of directories.

ls
jgs  U/

```

## # MISSION 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".
```

Introduciamo qui il comando per copiare i file: `cp [NOME_FILE][DESTINAZIONE]`

Portiamo a termine la missione copiando gli “standardi” dentro il nostro scrigno.

I comandi di selezione rapida visti in precedenza tornano ancora utili.

### *Esecuzione*

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

~/Castle
[mission 10] $ cd Great_hall

~/Castle/Great_hall
[mission 10] $ ls
3168_decorative_shield  7215_suit_of_armour  standard_2  standard_4
43277_stag_head         standard_1           standard_3

~/Castle/Great_hall
[mission 10] $ cp *standard* /home/kali/gameshell.1/World/Forest/Hut/Chest

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

Congratulations, mission 10 has been successfully completed!
```

## # MISSION 11

```
(0)===><=====
)
) 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.
)
) =====
(0)===><=====
```

Missione molto simile alla precedente, utilizziamo il comando `cp *tapestry*` per copiare tutti i file contenenti la parola "tapestry" nello scrigno

## Esecuzione

```
[mission 11] $ cd Castle
~/Castle
[mission 11] $ ls
Cellar/ Great_hall/ Main_building/ Main_tower/ Observatory/
~/Castle
[mission 11] $ cd Great_hall
~/Castle/Great_hall
[mission 11] $ ls
21222_tapestry_03      29764_stag_head      46552_suit_of_armour  62903_tapestry_09    standard_4
24006_decorative_shield 31801_tapestry_07    56322_tapestry_05    standard_1
24598_tapestry_06      32238_tapestry_10    57412_tapestry_02    standard_2
29383_tapestry_04      45113_tapestry_08    60358_tapestry_01    standard_3
~/Castle/Great_hall
[mission 11] $ cp *tapestry* /home/kali/gameshell.1/World/Forest/Hut/Chest
~/Castle/Great_hall
[mission 11] $ gsh check
Congratulations, mission 11 has been successfully completed!
```

## # MISSION 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.
)@==(
```

Questa missione introduce il comando `ls` con il flag `-l`.

Il comando `ls -l` permette di visualizzare il contenuto di una directory e alcuni dettagli degli stessi.

Tra questi dettagli abbiamo anche la data di creazione che ci permette di capire quale dei quadri è più “antico”.

Il secondo file è stato creato nel 1988, copiamolo quindi nello scrigno

### Esecuzione

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

~/Castle
[mission 12] $ ls
Cellar/ Great_hall/ Main_building/ Main_tower/ Observatory/

~/Castle
[mission 12] $ cd Main_tower

~/Castle/Main_tower
[mission 12] $ ls
First_floor/

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

~/Castle/Main_tower/First_floor
[mission 12] $ ls -l
total 16
-rw-rw-r-- 1 kali kali 1503 Mar 24 2014 painting_ceIdAgMc
-rw-rw-r-- 1 kali kali 1455 Sep 13 1988 painting_jhMfw0sV
-rw-rw-r-- 1 kali kali 1055 Apr 26 1995 painting_NuMaPHgu
drwxrwxr-x 3 kali kali 4096 Apr 18 14:48 Second_floor/

~/Castle/Main_tower/First_floor
[mission 12] $ cp painting_jhMfw0sV /home/kali/gameshell.1/World/Forest/Hut/Chest

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

Congratulations, mission 12 has been successfully completed!
```

## # MISSION 13

```
( ^ )----- ( ^ )
/ Mission goal
/ =====
/
/ Nostradamus predicted a spectacular star conjunction on the 08-16-1989.
/ 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.
( ^ )----- ( ^ )
```

Questa missione ci chiede di scoprire che giorno della settimana fosse il 16 agosto 1989.

Con il comando `cal 1989` il terminale stamperà il calendario completo del 1989.

Alla fine potremo rispondere che il 16 agosto era un mercoledì.

### Esecuzione

```
~/Castle/Main_tower/First_floor
[mission 13] $ cal 1989

      January          1989          March
Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7      1  2  3  4      1  2  3  4
 8  9 10 11 12 13 14    5  6  7  8  9 10 11    5  6  7  8  9 10 11
15 16 17 18 19 20 21   12 13 14 15 16 17 18   12 13 14 15 16 17 18
22 23 24 25 26 27 28   19 20 21 22 23 24 25   19 20 21 22 23 24 25
29 30 31              26 27 28              26 27 28 29 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                   1  2  3  4  5  6      1  2  3
 2  3  4  5  6  7  8    7  8  9 10 11 12 13    4  5  6  7  8  9 10
 9 10 11 12 13 14 15   14 15 16 17 18 19 20   11 12 13 14 15 16 17
16 17 18 19 20 21 22   21 22 23 24 25 26 27   18 19 20 21 22 23 24
23 24 25 26 27 28 29   28 29 30 31           25 26 27 28 29 30
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                   1  2  3  4  5      1  2
 2  3  4  5  6  7  8    6  7  8  9 10 11 12    3  4  5  6  7  8  9
 9 10 11 12 13 14 15   13 14 15 16 17 18 19   10 11 12 13 14 15 16
16 17 18 19 20 21 22   20 21 22 23 24 25 26   17 18 19 20 21 22 23
23 24 25 26 27 28 29   27 28 29 30 31       24 25 26 27 28 29 30
30 31

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

~/Castle/Main_tower/First_floor
[mission 13] $ gsh check
What was the day of the week for the 08-16-1989?
 1 : Monday
 2 : Tuesday
 3 : Wednesday
 4 : Thursday
 5 : Friday
 6 : Saturday
 7 : Sunday
Your answer: 3

Congratulations, mission 13 has been successfully completed!
```

## # MISSION 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'
Create a synonym for a string, that will stand
for a command.

)()==(
(@==()
```

Ci viene ora presentata la possibilità di abbreviare la scrittura dei comandi tramite un "alias"

La missione ci propone di creare un alias per il comando `ls -A`, dandogli il nome `la`.

Da questo momento quindi quando digiteremo `la`, ci verranno mostrati i contenuti nascosti di una cartella.

Il comando di alias è:

```
alias [NOME]='COMANDO'
```

es. `alias la='ls-A'`

### Esecuzione

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

~/Castle/Main_tower/First_floor
[mission 14] $ la
.nice_rock      painting_jhMfw0sV  Second_floor/
painting_ceIdAgMc painting_NuMaPHgu

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

Congratulations, mission 14 has been successfully completed!
```

## # MISSION 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
/
/ Remark: do not use Control-s or Control-z!
/
/-----(^)
```

In questa missione viene richiesto di creare un file di testo all'interno dello scrigno, utilizzando **nano**.

**nano** è un semplice editor di testo di Linux, per creare un file quindi dirigiamoci all'interno della cartella *Chest* e digitiamo:

```
nano journal.txt
```

Ci si presenterà una schermata come questa, dove possiamo scrivere qualsiasi appunto da salvare.

```
GNU nano 8.4      journal.txt *
Short message

[New File]

^G Help      ^O Write Out ^F Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^V Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

Da notare sono i comandi in fondo alla pagina che, con le combinazioni di tasti indicate permettono di salvare, uscire e altri comandi utili

```
[ Read 1 line ]
^G Help      ^O Write Out ^F Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^V Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

Salviamo il file e continuiamo.

### Esecuzione

```
~
[mission 15] $ cd Forest/Hut/Chest
~/Forest/Hut/Chest
[mission 15] $ nano journal.txt
~/Forest/Hut/Chest
[mission 15] $ gsh check
Congratulations, mission 15 has been successfully completed!
```





## # MISSION 17

```
(\ 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.

Tab-Tab
Pressing the "Tabulation" key twice successively shows a list
of possible completions.

(*)
\\
))
^
```

Questa è una missione a tempo.

Abbiamo 20 secondi per entrare nella tana della “regina dei ragni” ed eliminarla.

In questo caso è fondamentale l'utilizzo della funzione TAB per completare in automatico il nome dei file/directory

### Esecuzione

```
~/Castle/Cellar
[mission 17] $ cd .Lair_of_the_spider_queen\ rOuYoQMRDDbHjR gzBkqjFRnEllEakw/

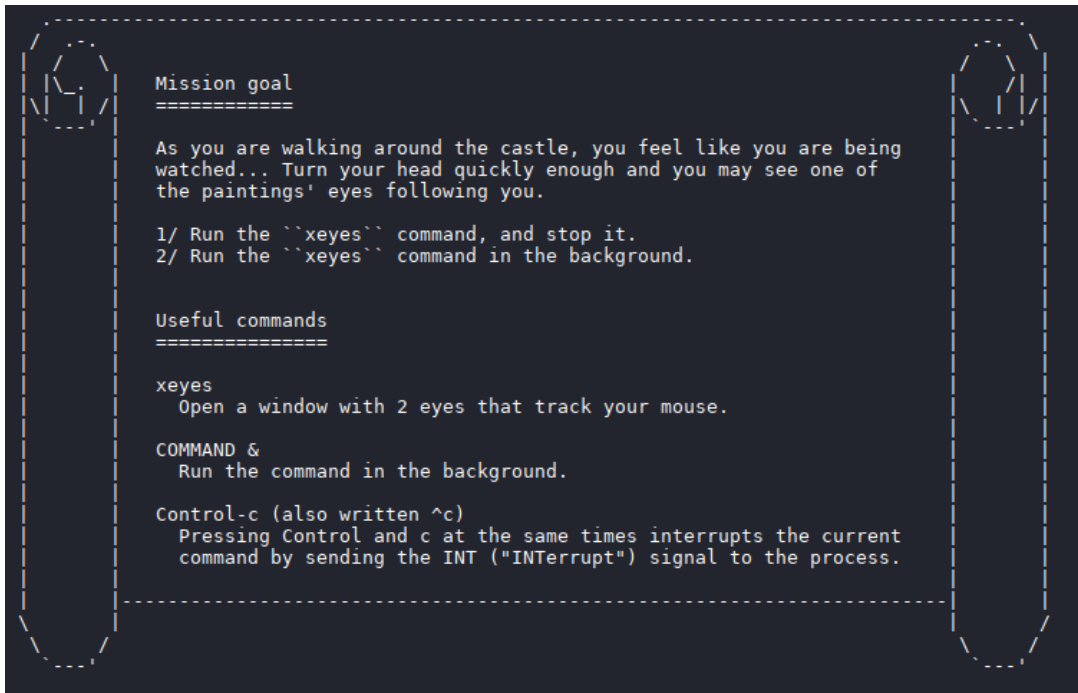
~/Castle/Cellar/.Lair_of_the_spider_queen rOuYoQMRDDbHjR gzBkqjFRnEllEakw
[mission 17] $ la
QEMYvcLqBUZRuvfS_spider_queen_uNmJbagopIOEVUXR  tcFnvFXyXYTYnii_baby_bat_OGNDnrDAYdmxYoYv

~/Castle/Cellar/.Lair_of_the_spider_queen rOuYoQMRDDbHjR gzBkqjFRnEllEakw
[mission 17] $ rm QEMYvcLqBUZRuvfS_spider_queen_uNmJbagopIOEVUXR

~/Castle/Cellar/.Lair_of_the_spider_queen rOuYoQMRDDbHjR gzBkqjFRnEllEakw
[mission 17] $ gsh check
Perfect, it took you only 20 seconds to complete this mission!

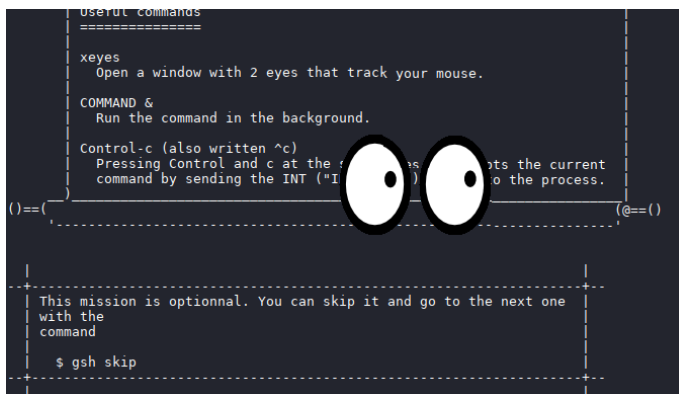
Congratulations, mission 17 has been successfully completed!
```

## # MISSION 18



Questa missione introduce i comandi per avviare e terminare un processo.

Digitando il nome del processo lo si avvia, in questo caso digitando `xeyes` compariranno questi occhi che seguiranno il movimento del mouse...

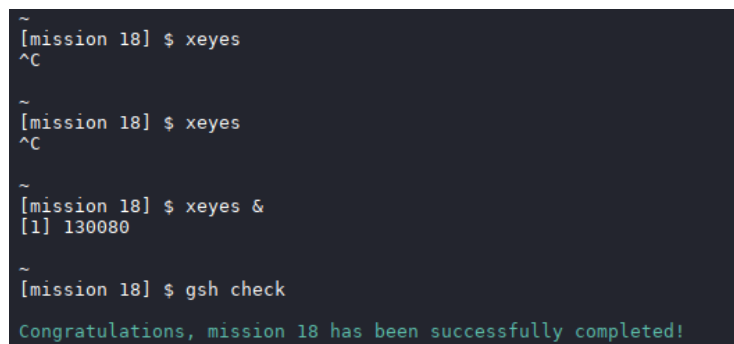


Per terminare il processo si usa la combinazione di tasti CNTRL + C

Aggiungendo una `&` alla fine del comando invece il processo si avvierà in background.

Notare come in questo caso ci viene restituita un numero, questo è il PID, un numero univoco assegnato da Linux ad ogni processo.

### Esecuzione



## # MISSION 19

Questa missione è una questione di tempismo.

Ci viene richiesto di far partire 3 “fuochi d'artificio” consecutivi, eseguibili con il comando **flarigo**, mentre il pirotecnico li guarda.

La richiesta è quindi di far partire i processi **flarigo** e il check del livello in parallelo.

Nel frattempo ho creato un alias del comando `gsh check`, abbreviandolo in `gc`

```

/&\ \
(      )
\ \  /
Mission goal
=====

The king's pyrotechnician appears next to you. He asks you to
fire **at least 3 consecutive fireworks** so he can see them from
far away.

A single firework can be created with the magical word

    flarigo

Useful commands
=====

flarigo
    This (non standard) command creates a single small firework.

COMMAND &
    Run the given command, but don't wait until it is finished to
    return.
    The command will run in the "background".

COMMAND1 ; COMMAND2 ; ... ; COMMANDN
    Run the given commands one after the other.
    Each command is run when the previous one is finished.

COMMAND1 & COMMAND2 & ... & COMMANDN
    Run the given commands "in parallel".
    All the commands are run in the "background", except the last
    one.
/&\ \
(      )
\ \  /

```

Per avviare processi in parallelo da terminale, basta concatenarli nella riga di comando con il carattere `&`, in un comando come:

```
gc & flarigo & flarigo & flarigo
```

## Esecuzione

[illegible]

## # MISSION 20

Questa missione consiste nel trovare la parola chiave di 4 lettere da accostare al processo **charmiglio** per il gran finale dei fuochi d'artificio.

Lo scopo è farci familiarizzare con il comando da tastiera `CNTRL+C`, che serve a stoppare i processi in esecuzione, dato che ad ogni tentativo, partirà un'animazione che darà il risultato dopo qualche secondo.

Per abbreviare il tutto interrompiamo i processi con `CNTRL+C` per avere subito il risultato del nostro tentativo.

```

(0)===>.....==>(0)
)
) Mission goal
) =====
)
) The king's pyrotechnician is trying to remember the magical
) incantation for creating the grand finale for his fireworks. This
) incantation starts with the word charmiglio and must be followed
) by four random letters, as in
)
)   $ charmiglio abcd
) or
)   $ charmiglio oops
)
) Help the pyrotechnician by finding 4 letters producing appropriate
) fireworks.
)
) NOTE: when the four letters are incorrect, the magical reaction
) can take a very long time. You need to interrupt it!
)
) It will probably take several tries before finding a combination
) of letters that works.
)
) Useful commands
) =====
)
) charmiglio CCCC
)   This (non standard) command creates some fireworks:
)   - if the four letters are valid, the fireworks will start
)     after a few seconds,
)   - if the four letters are not valid, the whole magical
)     reaction will go on for a long time.
)
) Control-c (also written ^c)
)   Pressing Control and c at the same times interrupts the current
)   command by sending the INT ("INTerrupt") signal to the process.
)
).....==>(0)
(0)===>.....==>(0)

```

Dopo svariati tentativi finiti tutti male...

```

      ^^^^^^^^
    /-----\
   |         |
   |         |
   |         |
   \_____/
     |||||
    .==|||.|=.
    -#%&%$#-
      || ; :|
     _____.-#&@#&$#~-._____

```

That's not working, try a different combination...

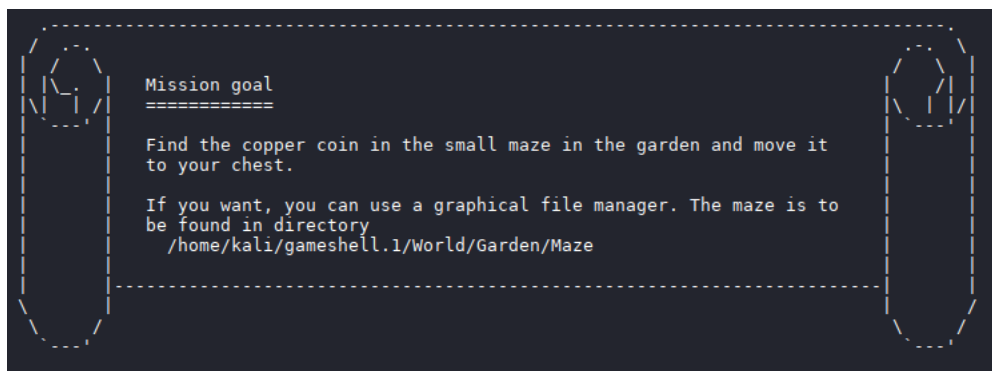
La combinazione vincente è: **charmiglio King**

```
It works! The special incantation is King
```

```
      .'.  
    : \ / :  
   '. \ / .' :  
  : \ / _ :  
 : \ / _ :  
: \ / _ :  
. '.  
 *  
  
      *  
     *  
    *  
   *  
  *  
 *
```

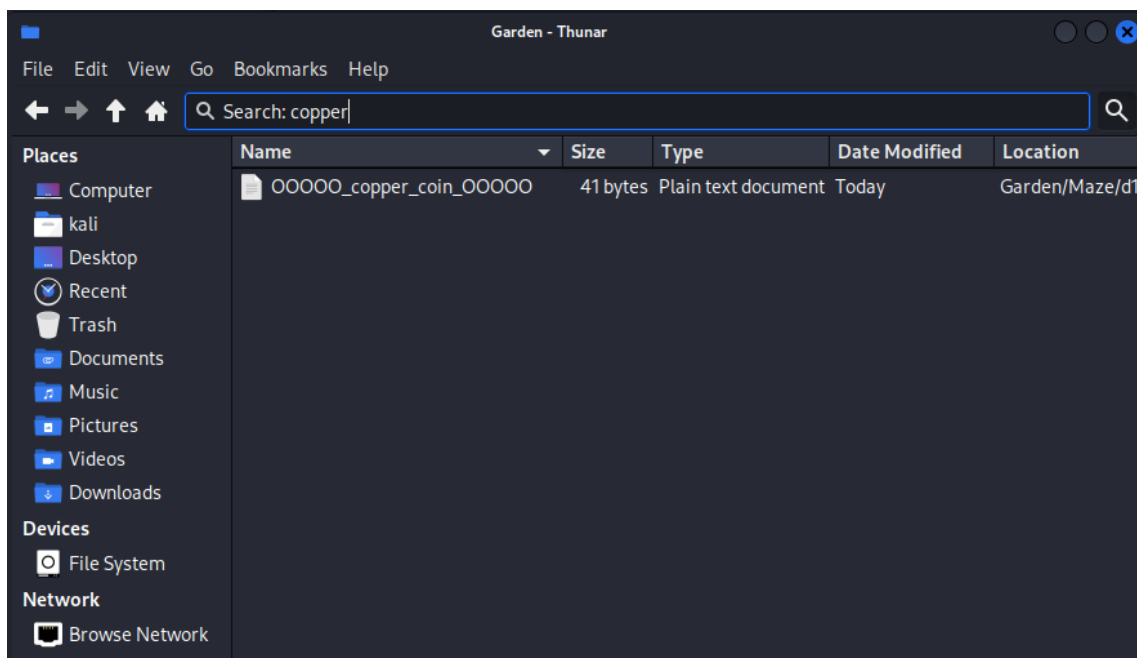
```
[mission 20] $ gc  
What's a valid 4 letters sequence? King  
  
Congratulations, mission 20 has been successfully completed!
```

## # MISSION 21



Dobbiamo cercare una moneta di bronzo all'interno di un labirinto

Il modo più veloce per superare questa missione (e come consiglia anche il gioco) è utilizzare l'interfaccia grafica di Linux e cercare la parola "copper" nella barra di ricerca, dato che il labirinto non è altro che un sistema di sottocartelle con nomi composti da caratteri casuali, quindi molto facile perdersi nei loro meandri (beh è un labirinto...)



Dopo di che basterà spostare il nella cartella del nostro scrigno per superare il livello.



UPGRADE: Da adesso tipologie di file differenti saranno visualizzati con colori differenti.

## # MISSION 22

```
{
  Mission goal
  =====

  Find the silver coin in the maze in the garden and move it to
  your chest using the shell.

  Useful commands
  =====

  ls -R
  Print the list of all files / directory, including those in
  sub-directories (recursively).

  tree
  Print the tree of files and directories, starting from the
  current working directory.
}
```

La missione è la stessa di prima, ma ora è richiesto esplicitamente che venga fatto tutto da terminale.

In nostro soccorso ci vengono presentati 2 comandi:

**ls -R** stampa la lista delle cartelle, sotto-cartelle e file dalla nostra posizione

**tree** stampa la lista di tutti i contenuti e li organizza in un albero

Utilizzando il comando **tree** ci viene mostrata questa “mappa del labirinto”, non ci resta che localizzare la moneta d’argento, raggiungerla e spostarla nello scrigno con il comando **mv**.

### Esecuzione

```
~/Garden/Maze/767f1408556a1cf/ffd73ff886/b412a2d79c3daeb94b6ff4d5376b
[mission 22] $ ls
00000_silver_coin_00000

~/Garden/Maze/767f1408556a1cf/ffd73ff886/b412a2d79c3daeb94b6ff4d5376b
[mission 22] $ mv 00000_silver_coin_00000 ~/Forest/Hut/Chest

~/Garden/Maze/767f1408556a1cf/ffd73ff886/b412a2d79c3daeb94b6ff4d5376b
[mission 22] $ gc

Congratulations, mission 22 has been successfully completed!
```

```
~/Garden/Maze
[mission 22] $ tree
.
├── 36cf9a169a6dc7069f3e98c
│   ├── 35bc25b7dc36eb8d
│   │   ├── 89e79f260ce5e39c8
│   │   ├── 98d00dd245eb
│   │   └── f0bc23896099ae
│   ├── 8ff4c196fece94966aaaae3d4
│   │   ├── 84edd95dde1a0f692178c2a6c
│   │   ├── 8cb58add47716382001c45a7c
│   │   └── a744c58e2c504ab763d9
│   └── eb8237a0a0b720903
│       ├── 7a95c02ddd910
│       ├── b93f94769eb436874bc42e771c
│       └── f6929f0b903122eb294d13f
├── 767f1408556a1cf
│   ├── 0384fe06ed87332ecabe330c5c123
│   │   ├── 0bfc6c704311030ea75fd9fc306
│   │   ├── ac4d580393ec7a3ef
│   │   └── ead7e4df3155563a446ac
│   ├── 3d47c3b14bd7e1
│   │   ├── 22427e49fc752987bfff177e4fb4c2
│   │   ├── 2407ef685abd0816fa19901a51
│   │   └── dc85829970590088176ff8a
│   └── ffd73ff886
│       ├── a5e10708258d5ae8ec556
│       ├── b412a2d79c3daeb94b6ff4d5376b
│       │   └── 00000_silver_coin_00000
│       └── fc747abbf6bd3becec51644342
├── 84ebcdbe65f01a200301e0e5c5c5b
│   ├── 42806eaabc98a78
│   │   ├── 7995d38e370c2982132cd65
│   │   ├── 8bb4695c72b332d87ab65167213
│   │   └── fe06a71a4d7f0f6eb17a
│   └── 4be67de4814c
│       ├── 32baae4a43f8c86a64613d0093
│       ├── 532a707cf72ea08c2cd5aeccc6ee34
│       ├── 96d92c4ec57a9a29ead59bc9a
│       └── b73e106ad4d92b9ce847
│           ├── 647425a07b385f99dcb4512f2f3c3f6
│           ├── 762653986252e768cd75d
│           └── da819060887217f8a58a930cddb4ad6
```

## # MISSION 23



Questa missione ci richiede di cercare ancora nel labirinto delle monete, stavolta d'oro.

Per farlo ci viene consigliato l'uso del comando `find`.

`find` è un comando molto potente della shell di Linux, che permette di cercare elementi con moltissimi filtri di ricerca, quali possono essere grandezza, data di creazione, nome del file, tipo ecc...

Questa volta il gioco non ci aiuta molto (volutamente) e ci invita a leggere il manuale del comando per imparare a padroneggiarlo.

Il manuale è molto lungo, ma dopo una ricerca sono riuscito a trovare una combinazione di pattern che mi permetteste di cercare la parola "gold" nei file delle sotto cartelle.

Il comando è il seguente: `find -iname "*gold*"`

Analizziamo il comando per capirlo più a fondo:

<code>find</code>	Comando di ricerca
<code>-iname</code>	Ricerca per nome, ignorando maiuscole e minuscole
<code>"*gold*"</code>	Cerca la stringa "gold" nei nomi dei file

```
~/Garden/Maze
[mission 23] $ find -iname "*gold*"
./ce3a47b671a2f9ed77c4b9eaa/6cfab340516f50119246832/376870bc1543360d974c255/gold_coin_1
./elc10a3d1e9f4091d18684f/056c2e28a6056f2/9fd1986e6b4ab5d92c788e7ce82/Gold_Coin_2
```

Una volta ottenuti i risultati, muoviamoci nei percorsi indicati e spostiamo i file, sempre con il comando `mv`

### Esecuzione

```
~/Garden/Maze/elc10a3d1e9f4091d18684f/056c2e28a6056f2/9fd1986e6b4ab5d92c788e7ce82
[mission 23] $ mv Gold_Coin_2 ~/Forest/Hut/Chest

~/Garden/Maze/elc10a3d1e9f4091d18684f/056c2e28a6056f2/9fd1986e6b4ab5d92c788e7ce82
[mission 23] $ gc

Congratulations, mission 23 has been successfully completed!
```



## # MISSION 24

```

Mission goal
=====

A forgetful old hermit called Servillus has set up camp in a cave with
his old, leather-bound potion book.
Go to the cave and help him remember the recipe of his famous herbal
tea.

In order to validate the mission, you need to be in the cave with
Servillus **and** your last command prior to ``gsh check`` must show the
recipe (including its title), but nothing else.

Note: you shouldn't alter the content of the book of potions.

Useful commands
=====

cat FILE
    Display the contents of the file.

head FILE
    Print the first 10 lines of the file.

head -n K FILE
    Print the first K lines of the file.

Remark
-----

A "FILE" may contain directories if the file in question is not in the
current directory.

```

In questa missione ci vengono presentati i comandi per leggere file:

cat Mostra il contenuto del file

`head` Mostra le prime 10 righe

`head -n K` Mostra K righe, che possiamo selezionare

La missione è un po' articolata, bisogna dirigersi nella cava e leggere solo le prime 6 righe del file richiesto, da quella posizione, di seguito l'esecuzione:

```
~/Mountain/Cave
[mission 24] $ cd

~
[mission 24] $ cd Mountain/Cave

~/Mountain/Cave
[mission 24] $ tree
.
├── Book_of_potions
│   ├── page_01
│   ├── page_02
│   ├── page_03
│   ├── page_04
│   ├── page_05
│   ├── page_06
│   ├── page_07
│   ├── page_08
│   ├── page_09
│   ├── page_10
│   ├── page_11
│   ├── page_12
│   ├── page_13
│   └── table_of_contents
└── servillus

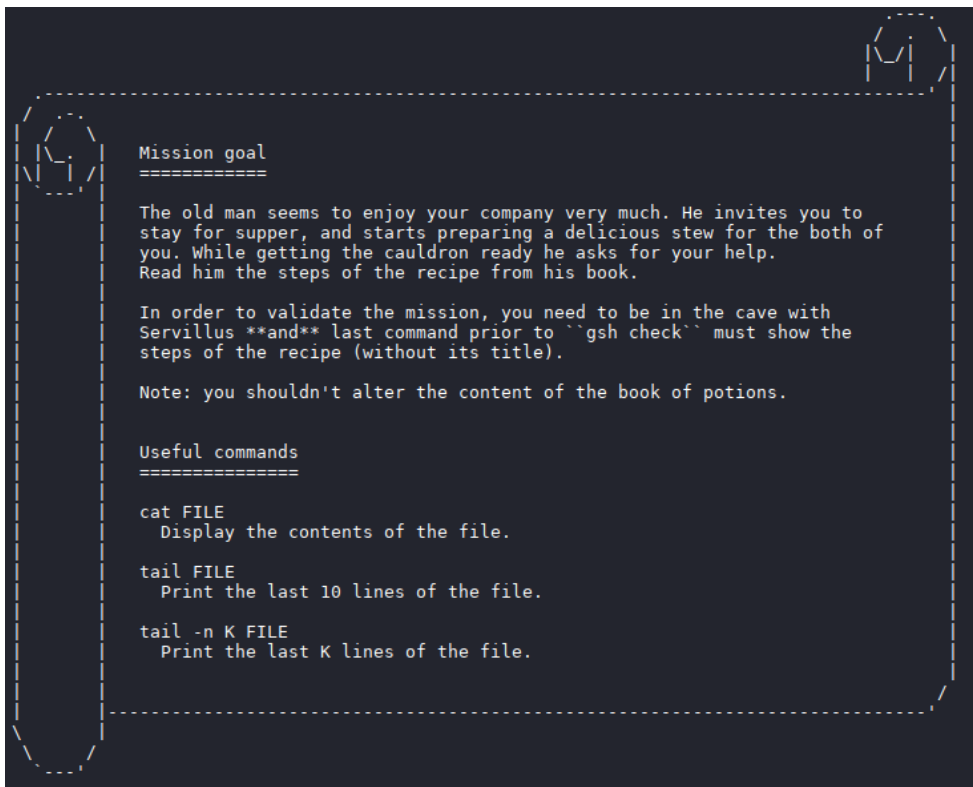
2 directories, 15 files

~/Mountain/Cave
[mission 24] $ cat Book_of_potions/table_of_contents
1. Transformation potion ..... pages 1-2
2. Elixir of youth ..... pages 3-4
3. Philter of love ..... page 5
4. Bottled death (powerful poison) ..... page 6
5. Herbal tea ..... page 7
6. Draft of invisibility ..... pages 7-8
7. Homeopathic healing potion (part 1) ..... pages 8-9
8. Homeopathic healing potion (part 2) ..... page 10
9. Homeopathic healing potion (part 3) ..... page 11
10. Toadstool stew ..... page 12
11. Distilled water ..... page 13
12. King's ale ..... page 13
```

Dopo aver visualizzato con `head` il file, e notato che andavano stampate solo le prime 6 righe, ho utilizzato il comando `head -n 6` sul file interessato per superare la missione.

[illegible]

## # MISSION 25



Missione molto simile alla precedente, con la differenza che dobbiamo stampare le ultime righe del file, operazione eseguibile con il comando `tail`

### Esecuzione

```
~/Mountain/Cave
[mission 25] $ tail -n 9 Book_of_potions/page_12
1) Boil water in a cauldron.
2) Add in a few death caps (Amanita phalloides).
3) Also add a few fly agarics (Amanita muscaria).
4) And some destroying angels (Amanita virosa).
5) Mix in a few deadly webcaps (Cortinarius rubellus).
6) Feel free to add in any colourful fungi you have on hand.
7) Let half of the water evaporate.
8) Season with a pinch of salt and a few herbs.
9) Serve hot in a bowl.

~/Mountain/Cave
[mission 25] $ gc

Congratulations, mission 25 has been successfully completed!
```

## # MISSION 26

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

While cleaning the dishes, Servillus mentions an interesting potion that
lets the drinker (temporarily) take the physical appearance of anyone.
Read the recipe of the potion from the hermit's book.

In order to validate the mission, you need to be in the cave with
Servillus **and** your last command prior to ``gsh check`` must show the
whole recipe (with its title).

Note: you shouldn't alter the content of the book of potions.

Useful commands
=====

cat FILE1 FILE2 ... FILEn
Display the contents of the files in order.
Remark: ``cat`` is an abbreviation for "concatenate".
)()==(
(@==(
```

Un'altra missione che ci mostra come visualizzare file.

Questa volta la ricetta da leggere è contenuta in 2 file differenti, per leggerla in una volta sola il comando `cat` permette di aprire più file contemporaneamente

```
cat FILE1 FILE2
```

### Esecuzione

```
~/Mountain/Cave
[mission 26] $ cat Book_of_potions/table_of_contents
1. Transformation potion ----- pages 1-2
2. Elixir of youth ----- pages 3-4
3. Philter of love ----- page 5
4. Bottled death (powerful poison) ----- page 6
5. Herbal tea ----- page 7
6. Draft of invisibility ----- pages 7-8
7. Homeopathic healing potion (part 1) ----- pages 8-9
8. Homeopathic healing potion (part 2) ----- page 10
9. Homeopathic healing potion (part 3) ----- page 11
10. Toadstool stew ----- page 12
11. Distilled water ----- page 13
12. King's ale ----- Page 13

~/Mountain/Cave
[mission 26] $ cat Book_of_potions/page_01 Book_of_potions/page_02
vvvvvvvvvvvvvvvvvvvvvv
Transformation potion
^^^^^^^^^^^^^^^^
1) Boil water in a cauldron.
2) Add 3 measures of fluxweed to the cauldron.
3) Add 2 bundles of knotgrass to the cauldron.
4) Stir 4 times, clockwise.
5) Wave your wand then let potion brew for 80 minutes.
6) Add 4 leeches to the cauldron.
7) Crush 2 scoops of lacewing flies to a fine paste.
8) Add 2 measures of the crushed lacewings to the cauldron.
9) Heat for 30 seconds on a low heat.
10) Add 3 measures of boomslang skin to the cauldron.
11) Crush a bicorn horn into a fine powder.
12) Add 1 measure of the crushed horn to the cauldron.
13) Heat for 20 seconds at a high temperature.
14) Wave your wand then let potion brew for 24 hours.
15) Add 1 additional scoop of lacewings to the cauldron.
16) Stir 3 times, counter-clockwise.
17) Split potion into multiple doses, if desired.
18) Add a pieces of the person you wish to become.
19) Wave your wand to complete the potion.

~/Mountain/Cave
[mission 26] $ gc

Congratulations, mission 26 has been successfully completed!
```

## # MISSION 27

```
( )------( )
/ Mission goal
/ =====
/
/ The old hermit notices your interest for potion recipes, and sees
/ promise in your ability to lookup lists of ingredients. He challenges
/ you to find the steps for the elixir of Youth.
/
/ In order to validate the mission, you need to be in the cave with
/ Servillus **and** your last command prior to ``gsh check`` must show the
/ steps for the recipe and nothing else.
/
/ Note: you shouldn't alter the content of the book of potions.
/
/ Useful commands
/ =====
/
/ cat FILE1 FILE2 ... FILEn
/   Display the contents of the files in order.
/
/ tail
/   Print the last 10 lines sent on the standard input.
/
/ tail -n K
/   Print the last K lines sent on the standard input.
/
/ COMMAND1 | COMMAND2
/   Run the two commands, feeding the "standard output" of the former into
/   the "standard input" of the latter.
/   Remark: by analogy with plumbing "|" is called "pipe".
/
/ Explanations
/ =====
/
/ Many of Unix commands process text: they receive text as input and
/ produce text as output.
/
/ It is common for those commands to write their output to their "standard
/ output", which means that (by default) the output is written into the
/ terminal.
/
/ Most of those commands can receive input either through files (given as
/ arguments) or from their "standard input". For example:
/ - ``head FILE`` reads its input from the file,
/ - ``head`` reads its input on the standard input.
/
/ By default, data from the standard input is read from the keyboard, but
/ a pipe can change that.
/------( )
```

Questa missione ci chiede ancora una volta di leggere delle pagine del “libro delle pozioni”, questa volta dovremmo eseguire due file contemporaneamente, ma stampando solo le ultime 7 righe del primo.

Per la soluzione ho utilizzato il comando:

```
tail -n 7 Book_of_the_potions/page_03 ; cat Book_of_the_potions/page_04
```

La presentazione della missione presentava la funzione “pipe”, che permette di passare l’output di un comando come input del comando successivo, ma sono riuscito comunque a risolverlo senza utilizzarlo, ma teniamolo presente: è un comando molto potente di cui sicuramente avremo bisogno in seguito.

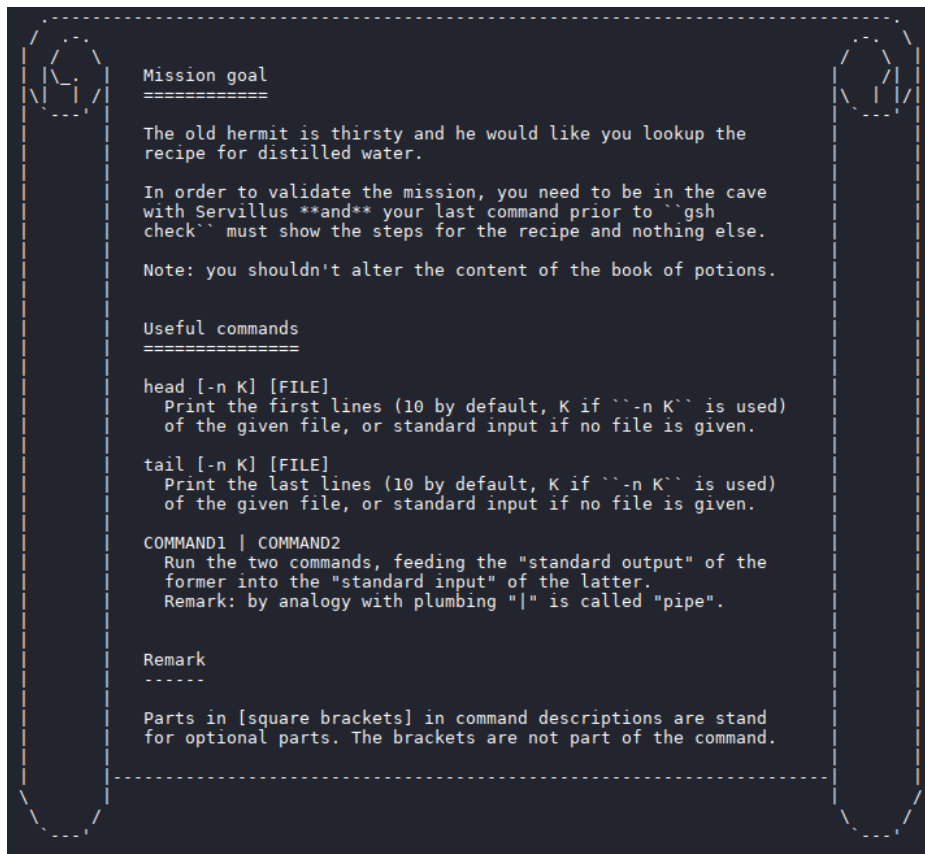
### Esecuzione

```
~/Mountain/Cave
[mission 27] $ tail -n 7 Book_of_potions/page_03 ; cat Book_of_potions/page_04
1) Fill a cauldron with used bath water.
2) Put a moderately large frog in the water.
3) Let the preparation rest overnight.
4) The next morning thank and free your little green friend.
5) Boil the water and add in a few sticks of oak tree.
6) Crush 5 river stones to a fine powder.
7) Mix in a third of the powder and stir vigorously.
8) Let the preparation rest for a day.
9) Add hairs from the tail of a squirrel (willingly given).
10) Add the remaining stone powder.
11) Stir the potion very vigorously, in all directions.
12) Take some time to rest after such an effort.
13) Rest a little bit more.
14) Even take a nap if you want.
15) Add a few larch tree needles for seasoning.
16) Drink the potion from the cauldron.

~/Mountain/Cave
[mission 27] $ gc

Congratulations, mission 27 has been successfully completed!
```

## # MISSION 28



Dobbiamo ancora visualizzare una ricetta, mostrando solo i passaggi e togliendo il resto, titolo compreso.

Questa volta utilizzeremo la funzione “pipe”:

```
head -n 6 Book_of_potions/page_13 | tail -n 3
```

Analizziamo il comando:

```
head -n 6 Book_of_potions/page_13
```

Restituisce le prime 6 righe del file page\_13

```
| tail -n 3
```

Prende in input le 6 righe precedenti e ne stampa solo le ultime 3

### ESECUZIONE

```

~/Mountain/Cave
[mission 28] $ cat Book_of_potions/table_of_contents
1. Transformation potion ----- pages 1-2
2. Elixir of youth ----- pages 3-4
3. Philter of love ----- page 5
4. Bottled death (powerful poison) ----- page 6
5. Herbal tea ----- page 7
6. Draft of invisibility ----- pages 7-8
7. Homeopathic healing potion (part 1) ----- pages 8-9
8. Homeopathic healing potion (part 2) ----- page 10
9. Homeopathic healing potion (part 3) ----- page 11
10. Toadstool stew ----- page 12
11. Distilled water ----- page 13
12. King's ale ----- Page 13

~/Mountain/Cave
[mission 28] $ head -n 6 Book_of_potions/page_13 | tail -n 3
1) Boil water in a big pot.
2) Condense the vapor in a fresh container.
3) Add minerals for a better taste (optional).

~/Mountain/Cave
[mission 28] $ gc

Congratulations, mission 28 has been successfully completed!

```

## # MISSION 29

```
( ^ ) ----- ( ^ )
/ Mission goal
/ =====
/ A mischievous imp cast a spell that puts smudges of coal
/ everywhere in the castle.
/ Find this spell and remove it.
/
/ Remark
/ -----
/ The spell is a process.
/
/ Useful commands
/ =====
/
/ ps
/ List the processes that are currently executed by the
/ shell.
/
/ kill N
/ Send the termination signal to process number N.
/ Remark: N is called PID, or "process identifier".
/
/ clear
/ Clear the screen.
/ The keybinding "Control-L" does the same and is often
/ quicker to use in the terminal.
( ) ----- ( )
```

```
~
[mission 29] $

      #@*
      & **/~
      !$-#

#@*
& **/~
!$-#

#@*
& **/~
!$-#

#@*
& **/~
!$-#

#@*
& **/~
!$-#

#@*
& **/~
!$-#

#@*
& **/~
!$-#
```

Un folletto dispettoso ha lanciato un incantesimo che sta macchiando tutto di carbone. Dobbiamo fermarlo!

Come spiega anche l'obiettivo della missione, "l'incantesimo" non è altro che un processo da terminare.

Visualizziamo quindi i processi con il comando **ps**

```
ps
  PID TTY          TIME CMD
356691 pts/0    00:00:00 zsh
356752 pts/0    00:00:00 bash
356811 pts/0    00:00:00 bash
357170 pts/0    00:00:00 spell
359162 pts/0    00:00:00 ps
```

Intercettato il PID del processo "spell" usiamolo per *killare* il processo con il comando:

```
kill 357170
```

### ESECUZIONE

```
kill 357170
bash: kill: 3571kill: arguments must be process or job IDs
bash: kill: kill: arguments must be process or job IDs
[1]+  Terminated                  "$GSH_TMP/${gettext "spell"}"

~
[mission 29] $ gc

Congratulations, mission 29 has been successfully completed!
```

## # MISSION 30

```
-----
/ Mission goal
/ =====
/
/ The mischievous imp has more than one trick up his sleeve. He
/ managed to protect his spell against most tampering.
/ You need to find this spell and try to remove it with standard
/ signal. If it doesn't work, use a more brutal signal.
/
/ Remark
/ -----
/
/ The spell is a process.
/
/ Useful commands
/ =====
/
/ ps
/ List the processes that are currently executed by the
/ shell.
/
/ kill [OPTIONS] N
/ Send the termination signal to process number N.
/
/ Useful options:
/ -s SIGNAL choose the signal name
/ -NUMBER   choose the signal number
/ -l        list available signals
/
/ clear
/ Clear the screen.
/ The keybinding "Control-L" does the same and is often
/ quicker to use in the terminal.
/
/ Details
/ -----
/
/ By default ``kill`` sends the "TERM" signal to the processes
/ (TERM stands for "termination").
/ Processes may ignore some signals, but the "KILL" signal
/ cannot be ignored!
/-----
```

Sembra che il comando `kill` non basti a fermare l'incantesimo del folletto. Serve qualcosa di più drastico.

Ci vengono quindi introdotti i SIGNAL del comando `kill`.

I SIGNAL sono dei comandi che possono essere usati per inviare segnali ad un processo.

Con il comando `kill -l` possiamo visualizzare la lista dei SIGNAL disponibili.

1) SIGHUP	2) SIGINT	3) SIGQUIT	4) SIGILL	5) SIGTRAP
6) SIGABRT	7) SIGBUS	8) SIGFPE	9) SIGKILL	10) SIGUSR1
11) SIGSEGV	12) SIGUSR2	13) SIGPIPE	14) SIGALRM	15) SIGTERM
16) SIGSTKFLT	17) SIGCHLD	18) SIGCONT	19) SIGSTOP	20) SIGTSTP
21) SIGTTIN	22) SIGTTOU	23) SIGURG	24) SIGXCPU	25) SIGXFSZ
26) SIGVTALRM	27) SIGPROF	28) SIGWINCH	29) SIGIO	30) SIGPWR
31) SIGSYS	34) SIGRTMIN	35) SIGRTMIN+1	36) SIGRTMIN+2	37) SIGRTMIN+3
38) SIGRTMIN+4	39) SIGRTMIN+5	40) SIGRTMIN+6	41) SIGRTMIN+7	42) SIGRTMIN+8
43) SIGRTMIN+9	44) SIGRTMIN+10	45) SIGRTMIN+11	46) SIGRTMIN+12	47) SIGRTMIN+13
48) SIGRTMIN+14	49) SIGRTMIN+15	50) SIGRTMAX-14	51) SIGRTMAX-13	52) SIGRTMAX-12
53) SIGRTMAX-11	54) SIGRTMAX-10	55) SIGRTMAX-9	56) SIGRTMAX-8	57) SIGRTMAX-7
58) SIGRTMAX-6	59) SIGRTMAX-5	60) SIGRTMAX-4	61) SIGRTMAX-3	62) SIGRTMAX-2
63) SIGRTMAX-1	64) SIGRTMAX			

Per la nostra missione ci serve qualcosa che arresti forzatamente il processo *spell*, quindi il SIGKILL, identificato con il numero 9

Dopo aver identificato il PID del processo *spell* quindi, inviamo questo comando di arresto forzato

```
kill -9 365531
```

```
kill -9 365531

~
[mission 30] $ gc

Congratulations, mission 30 has been successfully completed!
```



## # MISSION 31

```
( ^ )
-----
/ Mission goal
/ =====
/
/ The imp is comparing his magic with a fairy. They met in the
/ cellar, and imp is conjuring lumps of coal while the fairy is
/ conjuring delicate snowflakes.
/
/ Remove the imp's spells and the coal that litters the cellar,
/ but don't touch the snowflakes!
/
/ Remark
/ -----
/
/ Do not kill the imp or the fairy.
/
/ Useful commands
/ =====
/
/ pstree PID
/ Print the list of processes with their parent / child
/ relationship.
/
/ If no PID is given, show the list of all processes with
/ their parent / child relationship.
/
/ Useful options:
/ -p show the PID of processes
/ $$ This variable contains the PID of the
/ shell and can be given as the PID.
/
/ kill N
/ Send the termination signal to process number N.
/ Remark: N is called PID, or "process identifier".
/
( ^ )
```

Ancora questo folletto... adesso sta riempiendo le cantine di carbone.

Lo scopo è ancora quello di fermare i suoi incantesimi/processi, ma non quelli della fata.

In questa missione è fondamentale l'utilizzo del comando `pstree`, che restituisce un albero con le liste dei processi con la loro relazione padre/figlio.

Abbinato al flag `-p`, che mostra i PID è uno strumento molto utile per identificare i processi

Per svolgere la missione innanzitutto visualizziamo i processi attivi con `ps`

Abbiamo diversi processi *spell*, ma ricordiamoci che dobbiamo eliminare solo quelli che derivano dal folletto, quindi, identifichiamo il PID del processo *mischievous\_imp*, e lanciamo il comando:

```
pstree -p 371931
```

```
~/Castle/Cellar
[mission 31] $ pstree -p 371931
mischievous_imp(371931)─spell(371952)─sleep(465346)
                        └─spell(371953)─sleep(465258)
                        └─spell(371954)─sleep(465302)
                        └─tail(371960)
```

```
~/Castle/Cellar
[mission 31] $ ps
  PID TTY          TIME CMD
 356691 pts/0    00:00:00 zsh
 356752 pts/0    00:00:00 bash
 356811 pts/0    00:00:00 bash
 371929 pts/0    00:00:00 nice_fairy
 371931 pts/0    00:00:00 mischievous_imp
 371939 pts/0    00:00:00 tail
 371952 pts/0    00:00:00 spell
 371953 pts/0    00:00:00 spell
 371954 pts/0    00:00:00 spell
 371960 pts/0    00:00:00 tail
 462152 pts/0    00:00:00 sleep
 462196 pts/0    00:00:00 sleep
 462240 pts/0    00:00:00 sleep
 462249 pts/0    00:00:00 ps
```

Ora abbiamo una chiara panoramica dei collegamenti tra i processi.

Terminiamo i processi:

```
~/Castle/Cellar
[mission 31] $ kill 371952 371953 371954
```

Ripuliamo le cantine dal carbone, e la missione è compiuta.

```
~/Castle/Cellar
[mission 31] $ rm *coal*

~/Castle/Cellar
[mission 31] $ gc

Congratulations, mission 31 has been successfully completed!
```



1. *Journal of Management Studies*, 1997, 34, 1, 1-14.

\_\_\_\_\_

## # MISSION 33

```
(\
Mission goal
=====

To get better in the magical art, one needs to know mental
math.

Get ready, because Merlin is about to test your speed with
products.

Run the command ``gsh check`` to start.

Remark
-----

There now is a time constraint.

Hint
----

The library is rumored to contain some mathematics books and
hidden volumes.

Useful commands
=====

COMMAND < FILE
  Replace the command's standard input by a file.
  Instead of reading lines from the keyboard device, the
  command will read lines from the file.
\
(*)
))
^
```

Questa missione sembra matematica come la precedente, ma non lo è.

Vengono proposti calcoli più complessi con un limite di tempo, bisogna trovare un modo per risolverli in automatico.

La guida del livello ci dà un suggerimento riguardo un certo “libro di matematica” nella biblioteca.

Spostiamoci quindi nella biblioteca, dove troviamo il file del “libro di matematica”

```
~/Castle/Main_building/Library
[mission 33] $ ls
Greek_Latin_and_other_modern_languages  Mathematics_101  Merlin_s_office/
```

Ora utilizziamo il consiglio sui comandi utili:

Se al comando `gsh check` (o `gc`), come input, invece che i miei inserimenti da tastiera, inserissi il file del libro di matematica?

Proviamo

```
gc < Mathematics_101
```

```
~/Castle/Main_building/Library
[mission 33] $ gc < Mathematics_101
90 * 25 = ?? 43 * 93 = ?? 51 * 80 = ?? 45 * 13 = ?? 76 * 78 = ?? 30 * 82 = ?? 47 * 73 = ?? 4
7 * 16 = ?? 68 * 99 = ?? 7 * 50 = ?? 83 * 5 = ?? 55 * 47 = ?? 32 * 62 = ?? 7 * 53 = ?? 1 * 8
4 = ?? 4 * 48 = ?? 79 * 67 = ?? 77 * 22 = ?? 100 * 8 = ?? 54 * 49 = ?? 65 * 46 = ?? 18 * 76
= ?? 54 * 55 = ?? 20 * 96 = ?? 100 * 89 = ?? 85 * 74 = ?? 28 * 41 = ?? 45 * 27 = ?? 81 * 41
= ?? 47 * 21 = ?? 20 * 27 = ?? 24 * 31 = ?? 32 * 44 = ?? 87 * 2 = ?? 17 * 43 = ?? 39 * 71 =
?? 62 * 19 = ?? 73 * 55 = ?? 96 * 45 = ?? 96 * 38 = ?? 55 * 87 = ?? 40 * 40 = ?? 43 * 90 = ?
? 75 * 44 = ?? 43 * 49 = ?? 43 * 76 = ?? 75 * 37 = ?? 29 * 8 = ?? 61 * 15 = ?? 7 * 30 = ?? 3
* 95 = ?? 86 * 51 = ?? 8 * 70 = ?? 85 * 23 = ?? 6 * 73 = ?? 33 * 31 = ?? 30 * 40 = ?? 6 * 6
5 = ?? 36 * 93 = ?? 50 * 92 = ?? 72 * 98 = ?? 64 * 32 = ?? 4 * 47 = ?? 52 * 65 = ?? 72 * 99
= ?? 77 * 2 = ?? 47 * 98 = ?? 50 * 7 = ?? 79 * 12 = ?? 2 * 38 = ?? 85 * 93 = ?? 33 * 31 = ??
66 * 36 = ?? 87 * 96 = ?? 9 * 32 = ?? 30 * 86 = ?? 13 * 50 = ?? 19 * 46 = ?? 71 * 61 = ?? 2
* 30 = ?? 4 * 23 = ?? 30 * 12 = ?? 41 * 51 = ?? 79 * 27 = ?? 20 * 47 = ?? 31 * 55 = ?? 7 *
42 = ?? 60 * 74 = ?? 86 * 1 = ?? 62 * 76 = ?? 85 * 7 = ?? 79 * 19 = ?? 59 * 40 = ?? 97 * 74
= ?? 4 * 15 = ?? 73 * 38 = ?? 11 * 51 = ?? 69 * 55 = ?? 75 * 87 = ?? 56 * 33 = ??
Congratulations, mission 33 has been successfully completed!
```

All'interno del file erano presenti i risultati delle operazioni richieste, che quindi prese come input hanno risolto le varie operazione consecutivamente.

## # MISSION 34

```

/&\ \
(   )
\_/\ \

Mission goal
=====

Merlin's old spell books are kept in his office, in the
library. You need to save a list of all those spell books
(and nothing else) in a file called "inventory.txt", in the
drawer...

Useful commands
=====

COMMAND > FILE
  Send the command's output to a file instead of printing it
  on the screen.

less FILE
  display the content of a file, one page at a time

  Important keyboardings are
    q          quit
    Space      scroll down one page
    / STRING   search for a string
    n          go to the next occurrence of the
               search string

ls FILE1 ... FILEn
  Show the list of files given as arguments.
  This is particularly useful if you use shell patterns with
  wildcards.

/&\ \
(   )
\_/\ \

```

In questa missione ci viene richiesto di copiare il nome di una lista di file all'interno di un file di testo *inventory.txt* dentro una determinata cartella (*Drawer*).

```
~/Castle/Main_building/Library/Merlin_s_office
[mission 34] $ ls
candle          grimoire_16658  grimoire_21617  grimoire_27299  grimoire_32723
Drawer/         grimoire_1698   grimoire_21663  grimoire_27720  grimoire_3347
grimoire_10372  grimoire_17218  grimoire_21736  grimoire_280    grimoire_3920
grimoire_10783  grimoire_17292  grimoire_22282  grimoire_28269  grimoire_4419
grimoire_11307  grimoire_17300  grimoire_22653  grimoire_28460  grimoire_5359
grimoire_11315  grimoire_17385  grimoire_22838  grimoire_2881   grimoire_5965
grimoire_11474  grimoire_17612  grimoire_23177  grimoire_29068  grimoire_5982
grimoire_11891  grimoire_17637  grimoire_23425  grimoire_29248  grimoire_6108
grimoire_12145  grimoire_17797  grimoire_2351   grimoire_2927   grimoire_6203
grimoire_12565  grimoire_17905  grimoire_23545  grimoire_29393  grimoire_6279
grimoire_12680  grimoire_17950  grimoire_23577  grimoire_29484  grimoire_6346
grimoire_13548  grimoire_18030  grimoire_23802  grimoire_30165  grimoire_6822
grimoire_13704  grimoire_18399  grimoire_24394  grimoire_30248  grimoire_7575
grimoire_13964  grimoire_18902  grimoire_25036  grimoire_30538  grimoire_8939
grimoire_14132  grimoire_1938   grimoire_25181  grimoire_30758  grimoire_9272
grimoire_14860  grimoire_20494  grimoire_25731  grimoire_30763  grimoire_9514
grimoire_15362  grimoire_20899  grimoire_26153  grimoire_31732  grimoire_9548
grimoire_15597  grimoire_2095   grimoire_26554  grimoire_31750  grimoire_9744
grimoire_16085  grimoire_21220  grimoire_26575  grimoire_3191   nano
grimoire_16625  grimoire_21309  grimoire_26581  grimoire_32210
grimoire_16643  grimoire_21582  grimoire_26760  grimoire_32347
```

Creiamo innanzitutto il file *inventory.txt* nella directory *Drawer*.

```
touch Drawer/inventory.txt
```

Utilizziamo quindi il seguente comando:

```
ls *grimoire* > Drawer/inventory.txt
```

In questo modo manderemo come input al file di testo, l'output del comando `ls` e possiamo concludere la missione

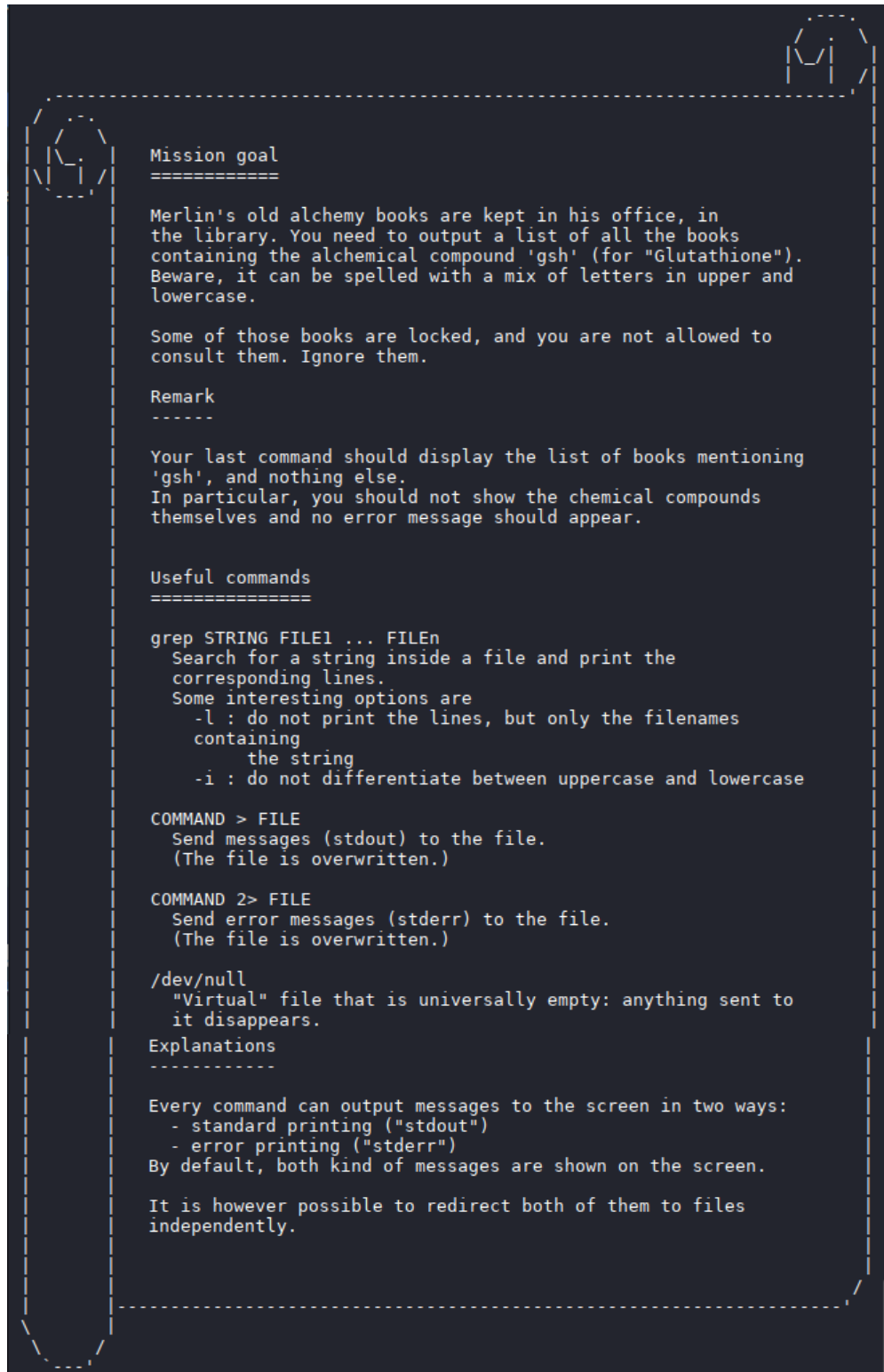
## ESECUZIONE

```
~/Castle/Main_building/Library/Merlin_s_office
[mission 34] $ ls *grimoire* > Drawer/inventory.txt

~/Castle/Main_building/Library/Merlin_s_office
[mission 34] $ gc

Congratulations, mission 34 has been successfully completed!
```

## # MISSION 35



In questa missione dovremmo creare un file di testo con all'interno i nomi dei file di una lista, che hanno al loro interno la parola "gsh" (ignorando maiuscole/minuscole).

Per questa missione dobbiamo utilizzare due nuovi comandi molto utili:

**grep STRING** Cerca i file con all'interno la stringa voluta

**COMMAND 2 >** Restituisce come output eventuali errori del comando

Torna utile anche l'utilizzo del percorso **/dev/null**, un file virtuale che permette di far "sparire" tutto ciò che riceve in input.

Per eseguire l'operazione con un solo comando (come richiede la missione) questa è la soluzione.

```
grep -li "gsh" * > books_list.txt 2> /dev/null ; cat books_list.txt
```

Scomponiamola e analizziamola:

<pre>grep -li "gsh" *</pre>	Cerca nei file della directory corrente il termine " <i>gsh</i> ", ignorando maiuscole e minuscole, e stampa solo i nomi dei file che lo contengono.
<pre>&gt; books_list.txt</pre>	Reindirizza l'output al file <i>output.txt</i> .
<pre>2&gt; /dev/null</pre>	Nasconde eventuali messaggi di errore.
<pre>; cat books_list.txt</pre>	Se il comando precedente ha successo, utilizza <code>cat</code> per stampare il contenuto di <i>output.txt</i> .

## ESECUZIONE

```
~/Castle/Main_building/Library/Merlin_s_office
[mission 35] $ grep -li "gsh" * > books_list.txt 2> /dev/null ; cat books_list.txt
grimoire_dNrRbxtdRSliecUj
grimoire_EiPyUQDCtMYIhDZizkreFjYUPB
grimoire_EqBkqHSmJG
grimoire_jrzluXrttTYuwyehmFGKSvlMKZKrqJh
grimoire_KaIbbUjhCvCHFe
grimoire_kGrEpuTBDJkzhypXVnjsQ
grimoire_ndHREffRXsVAA
grimoire_oeYlTaseKspByiORFoEcmtiV
grimoire_PH0dptcynBAsNjQz
grimoire_qiCoXvcfsLgcq0sGLAAFA
grimoire_rdCLfAuZSvkkUZnImpXaRkpV
grimoire_RGfztsFT
grimoire_SbybglCcWnItCSfUPRXkrqbQEaxyCg
grimoire_tznaSgxcuksCQySjvCCUZ
grimoire_uGuxyVz0duftGnNJTAVYsh
grimoire_vDOTXDrtYIgronaP
grimoire_vGmMsvSURw
grimoire_vQtihusHS0vqPWLwlZglStKwzWj0
grimoire_wgJrVLAfcer
grimoire_wPxnewLo0qAGlntzHqGchConL
grimoire_wvFNBAFASbECHQrepT
grimoire_wwhXLGaOuWaIyGPMd
grimoire_xvxQIhrXYTilJeLI
grimoire_YLqGPfQFBzHzXXjENyZc
grimoire_zbsxzHriZATV0dDuFclgEaNcAsHbyiF
grimoire_zrLFrIuLtbZ

~/Castle/Main_building/Library/Merlin_s_office
[mission 35] $ gc

Congratulations, mission 35 has been successfully completed!
```

## # MISSION 36



In questa missione dobbiamo decifrare il messaggio di Merlino per trovare la chiave segreta per superare il livello.

Viene suggerito l'uso del reindirizzamento, quindi proseguiamo reindirizzando l'incompressibile output di Merlino (che non è altro che un file eseguibile) ad un file di testo eliminando gli errori.

```
~/Castle/Observatory
[mission 36] $ ./merlin > key.txt
RzyAtzjWyigBxTjjn0kgUVMkp0FUBttXUYzbFikxLVbVyFBjAunMD
FXgBQFVG0VNgxSQIDwcMFjbgfYaHLwLJPEFJQqWUakrknEamGPgJX

~/Castle/Observatory
[mission 36] $ cat key.txt
THESECRETKEYISONSTDERR
```

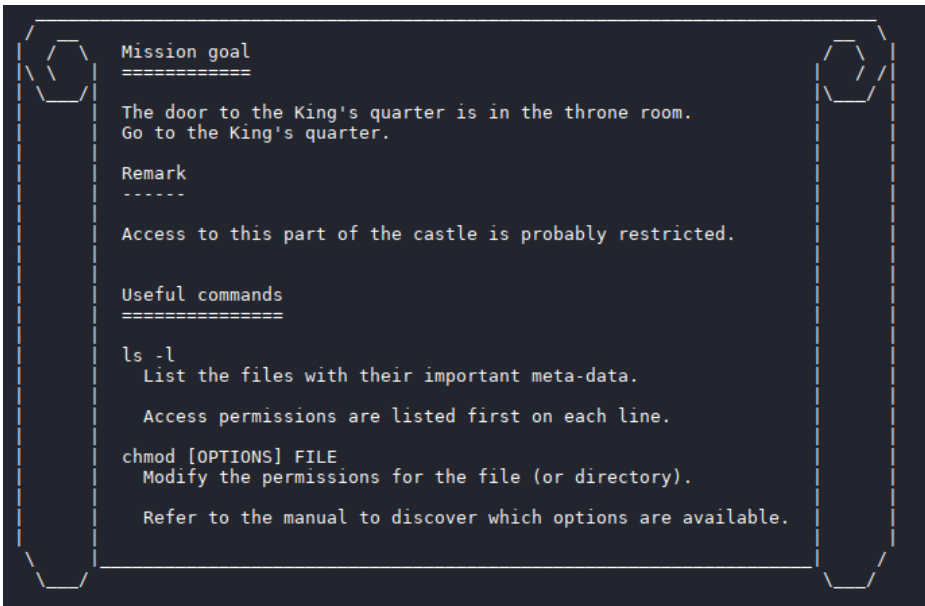
Sembrava troppo facile... Visualizzando il file appena salvato ci viene dato un messaggio: La chiave si trova nell'output di errore. Ripetiamo l'operazione ma stavolta salviamo sul file l'output di errore scartato prima.

```
~/Castle/Observatory
[mission 36] $ ./merlin 2> key.txt
THESECRETKEYISONSTDERR
```

Ora reindirizziamo il file con la chiave segreta al comando di check di fine livello.

```
~/Castle/Observatory
[mission 36] $ gc < key.txt
What is the secret key?
Congratulations, mission 36 has been successfully completed!
```

## # MISSION 37



In questa missione dobbiamo accedere ad una stanza in cui non abbiamo il permesso di entrare.

Vengono introdotti permessi sui file di Linux e il comando `chmod` che permette di modificarli.

I permessi di ogni file sono divisi per tipo di utente:

Ogni file ha 3 tipi di permessi:

- r** permesso di lettura
- w** permesso di scrittura
- x** permesso di esecuzione

E ognuno di questi permessi è modificabile in base alla tipologia di utente che ne ha accesso:

- U** permessi dell'utente
- G** permessi del gruppo
- O** permessi per tutti gli altri

Quindi ogni file ha assegnata una lista di 9 caratteri (3 tipi di permessi per ogni tipo di utente) attivabili e disattivabili, con il comando `chmod`

Per visualizzare i permessi di ogni file all'interno di una directory si usa il comando `ls -l`

Per modificare quindi i permessi della cartella *Kings\_quarter* ho utilizzato il comando:

`chmod 777 Kings_quarter`

777 è un modo rapido per concedere tutti i permessi a tutti i tipi di utente

```
~/Castle/Main_building
[mission 37] $ cd Throne_room/

~/Castle/Main_building/Throne_room
[mission 37] $ ls -l
total 4
drw-rw-r-- 2 kali kali 4096 Apr 18 14:48 Kings_quarter/

~/Castle/Main_building/Throne_room
[mission 37] $ chmod 777 Kings_quarter/

~/Castle/Main_building/Throne_room
[mission 37] $ ls -l
total 4
drwxrwxrwx 2 kali kali 4096 Apr 18 14:48 Kings_quarter/

~/Castle/Main_building/Throne_room
[mission 37] $ cd Kings_quarter/

~/Castle/Main_building/Throne_room/Kings_quarter
[mission 37] $ gc

Congratulations, mission 37 has been successfully completed!
```

## # MISSION 38

```
(\ Mission goal
=====

The King is rumored to keep the combination to his safe written on a note in his room.
Find that combination.

Remark
-----

The King probably tried to make this note unreadable!

Useful commands
=====

ls -l
  Lists the files with their important meta-data.

  Access permissions are listed first on each line.

chmod [OPTIONS] FILE
  Modifies the permissions for the file (or directory).

  Refer to the manual to discover which options are available.

(*)
))
^
```

Si vocifera che il Re abbia una combinazione in una nota nella sua stanza, dobbiamo trovarla.

Dalla stanza del re eseguo `ls -la` per avere la lista di tutti i file, anche nascosti, all'interno della directory, e dei loro permessi.

Trovate le note segrete abilito tutti i permessi, con un punto davanti al nome del file, dato che era nascosto.

Reindirizzo il file al processo di check e completo la missione.

### ESECUZIONE

```
~/Castle/Main_building/Throne_room/Kings_quarter
[mission 38] $ ls -la
total 16
drwxrwxr-x 2 kali kali 4096 Apr 20 12:54 ./
drwxrwxr-x 3 kali kali 4096 Apr 18 14:48 ../
-rwxrwxrwx 1 kali kali  11 Apr 20 12:54 note
--w--w---- 1 kali kali  11 Apr 20 12:54 .secret_note

~/Castle/Main_building/Throne_room/Kings_quarter
[mission 38] $ chmod 777 .secret_note

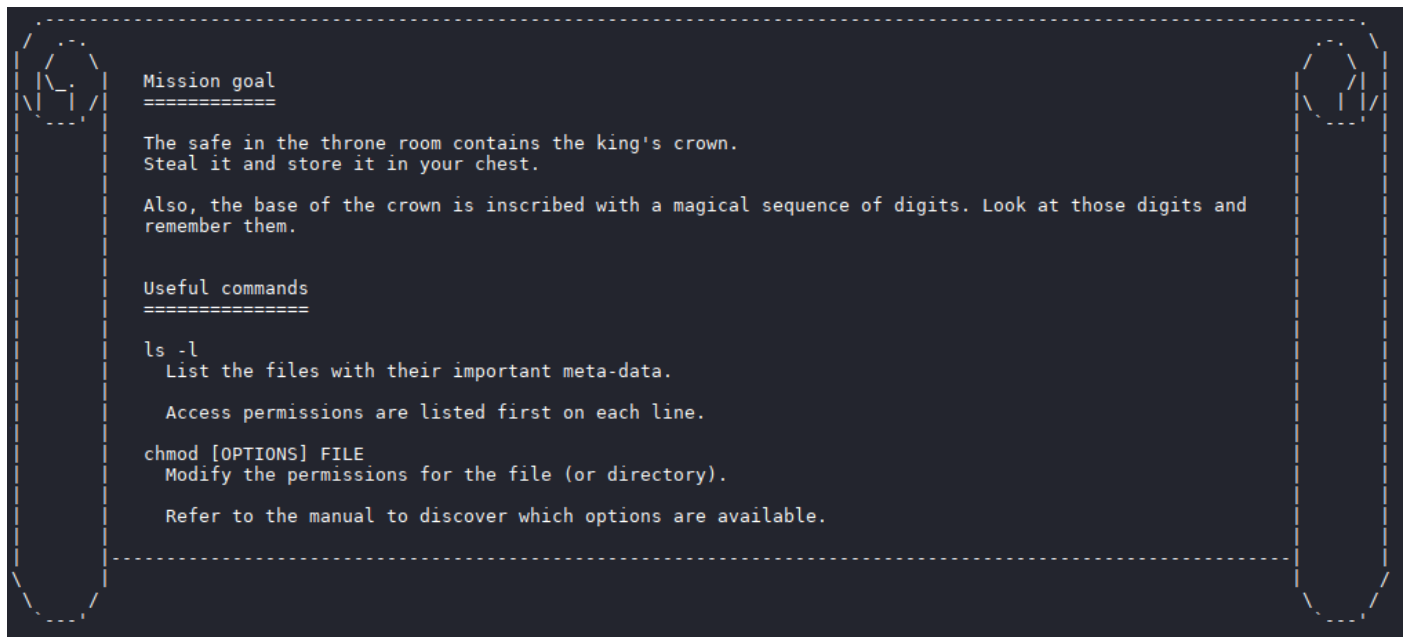
~/Castle/Main_building/Throne_room/Kings_quarter
[mission 38] $ ls -la
total 16
drwxrwxr-x 2 kali kali 4096 Apr 20 12:54 ./
drwxrwxr-x 3 kali kali 4096 Apr 18 14:48 ../
-rwxrwxrwx 1 kali kali  11 Apr 20 12:54 note
-rwxrwxrwx 1 kali kali  11 Apr 20 12:54 .secret_note

~/Castle/Main_building/Throne_room/Kings_quarter
[mission 38] $ cat .secret_note
3845425796

~/Castle/Main_building/Throne_room/Kings_quarter
[mission 38] $ gc < .secret_note
What's the combination to open the King's safe?
Congratulations, mission 38 has been successfully completed!
```



## # MISSION 39



Anche in questa missione dovremo abilitare i permessi di file/cartelle per poterle utilizzare.

Questa volta dobbiamo rubare la corona del Re dalla sua stanza *Safe*

Utilizzo sempre il comando `chmod 777`, seguito dal nome del file per abilitare i permessi, sia nella cartella *Safe* e sul file *crown*.

Visualizzo il file *crown* per memorizzare le cifre alla sua base, come ci dice il testo della missione, sposto il file dentro lo scrigno e supero la missione, digitando le tre cifre lette precedentemente.

### ESECUZIONE

```
~/Castle/Main_building/Throne_room
[mission 39] $ ls -la
total 16
drwxrwxr-x 4 kali kali 4096 Apr 20 12:58 ./
drwxrwxr-x 4 kali kali 4096 Apr 18 14:48 ../
drwxrwxr-x 2 kali kali 4096 Apr 20 12:54 Kings_quarter/
d----- 2 kali kali 4096 Apr 20 12:58 Safe/

~/Castle/Main_building/Throne_room
[mission 39] $ chmod 777 Safe

~/Castle/Main_building/Throne_room
[mission 39] $ cd Safe/

~/Castle/Main_building/Throne_room/Safe
[mission 39] $ ls -la
total 12
drwxrwxrwx 2 kali kali 4096 Apr 20 12:58 ./
drwxrwxr-x 4 kali kali 4096 Apr 20 12:58 ../
----- 1 kali kali 48 Apr 20 12:58 crown

~/Castle/Main_building/Throne_room/Safe
[mission 39] $ chmod 777 crown

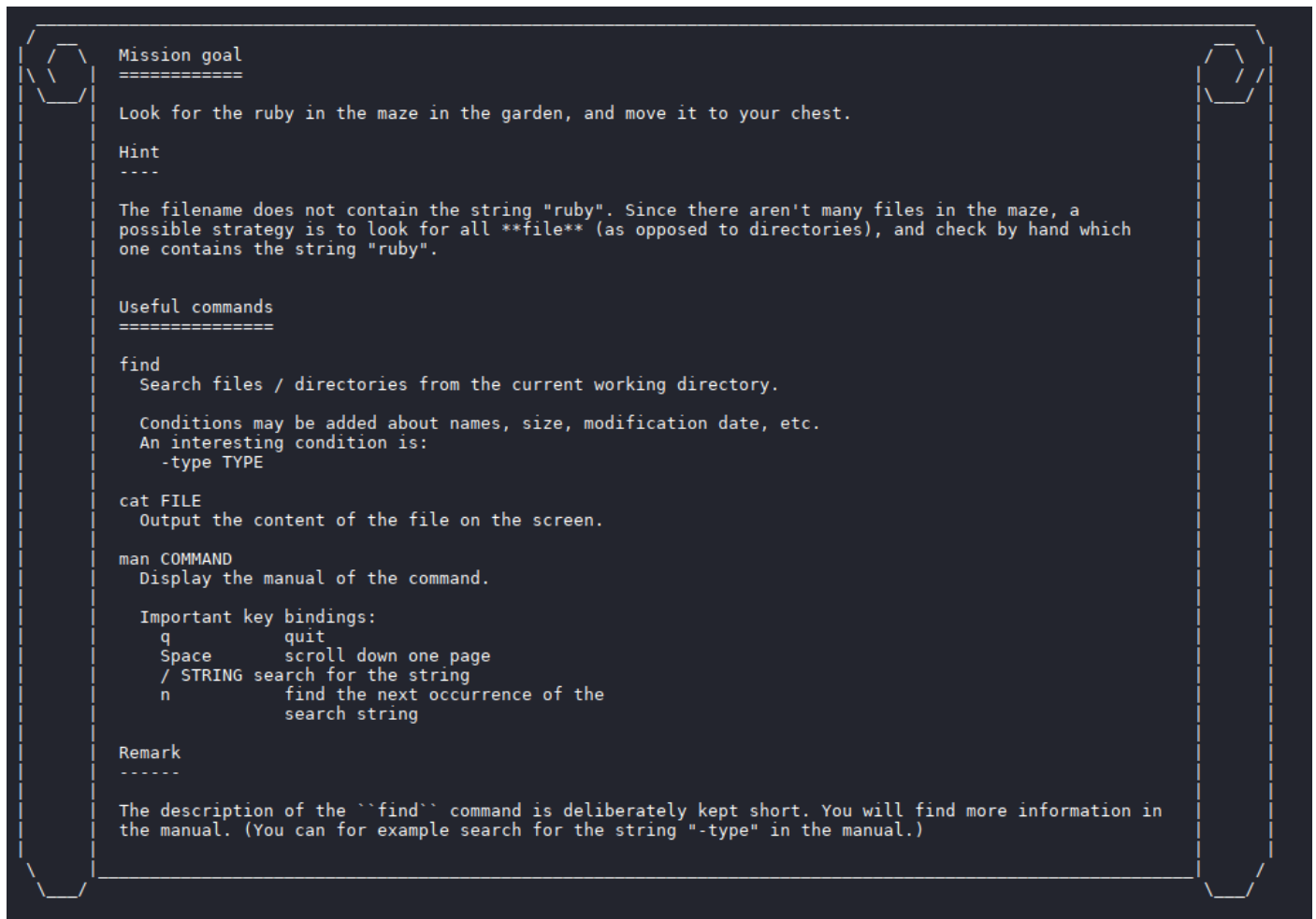
~/Castle/Main_building/Throne_room/Safe
[mission 39] $ cat crown
_+._
(^\/^\/^\/)
\@*@\@/
{_516_}

~/Castle/Main_building/Throne_room/Safe
[mission 39] $ mv crown ~/Forest/Hut/Chest/

~/Castle/Main_building/Throne_room/Safe
[mission 39] $ gc
What are the 3 digits inscribed on the base of the crown? 516

Congratulations, mission 39 has been successfully completed!
```

## # MISSION 40



Dirigiamoci di nuovo nel labirinto alla ricerca di un rubino.

La missione ci dice che cercare la parola “ruby” nel nome del file potrebbe non funzionare questa volta, e che la parola sarà contenuta all’interno del file.

Consigliamo di utilizzare il comando `find` per ottenere la lista dei file nella sequenza di sottocartelle, e poi esaminarli uno ad uno alla ricerca della parola “ruby”.

Velocizzeremo però il compito con questo comando:

```
find . -type f | xargs grep -l "ruby"
```

Analizziamolo:

<code>find . -type f</code>	Trova tutti i file (non directory) a partire dalla directory corrente.
<code> </code>	Passa l'output del comando <code>find</code> al comando successivo.
<code>xargs grep -l "ruby"</code>	Usa <code>xargs</code> per eseguire <code>grep</code> su ogni file trovato, stampando solo i nomi dei file che contengono la stringa "ruby".

`xargs` risulta fondamentale in questo caso, in quanto `grep` non può prendere direttamente l'output di `find` come argomento senza un intermediario. `find` stampa i nomi dei file su righe separate, e `grep` si aspetta di ricevere i nomi dei file come argomenti.

Una volta localizzato il file e copiato nello scrigno terminiamo la missione.

```
~/Garden/Maze
[mission 40] $ find . -type f | xargs grep -l "ruby"
./af00063038f837d/64ad38b66812ed46dc/86e6f10059bf004a5cfd/28282

~/Garden/Maze
[mission 40] $ cd af00063038f837d/64ad38b66812ed46dc/86e6f10059bf004a5cfd/

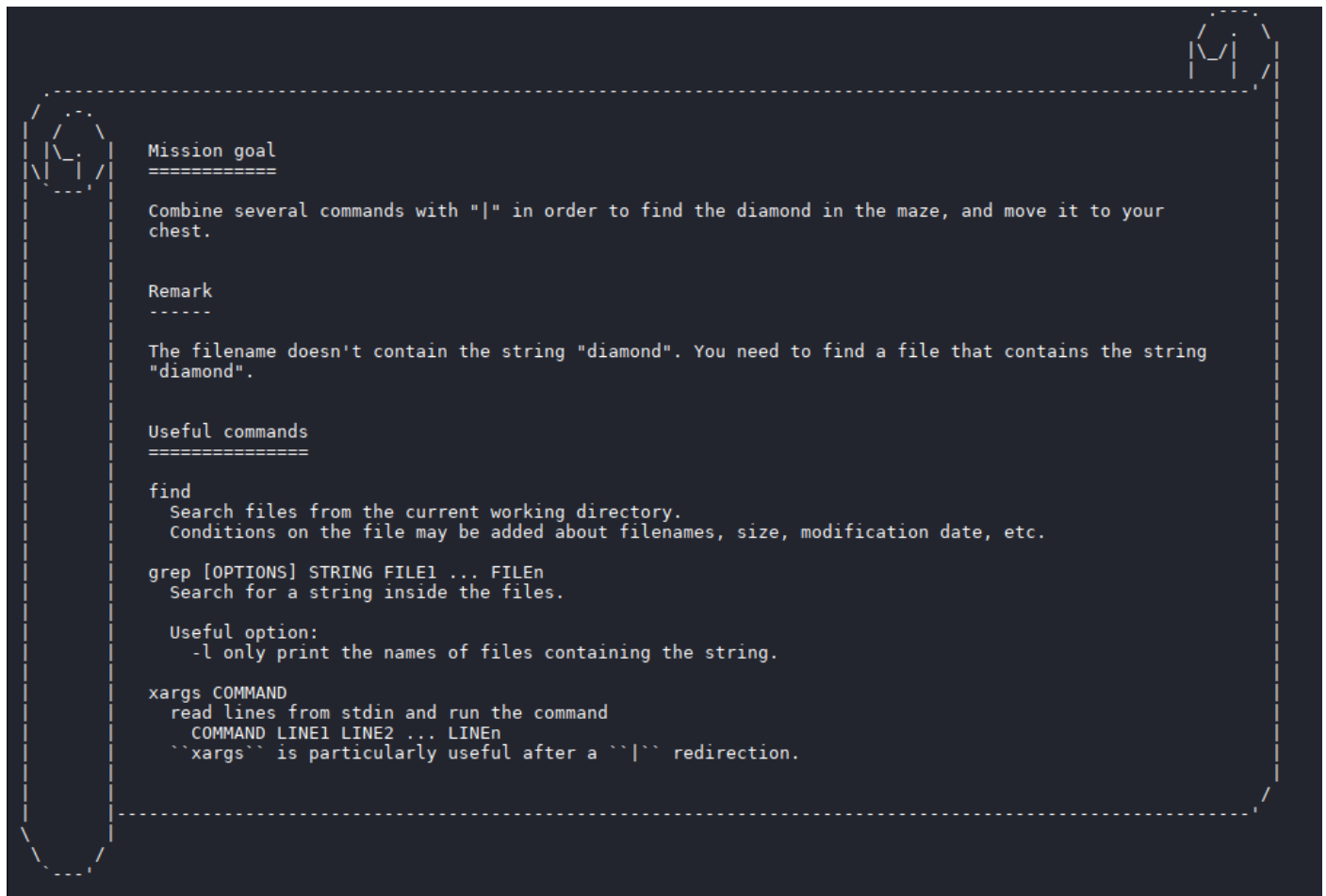
~/Garden/Maze/af00063038f837d/64ad38b66812ed46dc/86e6f10059bf004a5cfd
[mission 40] $ ls
28282

~/Garden/Maze/af00063038f837d/64ad38b66812ed46dc/86e6f10059bf004a5cfd
[mission 40] $ mv 28282 ~/Forest/Hut/Chest/

~/Garden/Maze/af00063038f837d/64ad38b66812ed46dc/86e6f10059bf004a5cfd
[mission 40] $ gc

Congratulations, mission 40 has been successfully completed!
```

## # MISSION 41



Ora dobbiamo cercare un diamante...

Nella missione precedente ho anticipato le intenzioni del gioco, ora è lui che ci invita ad utilizzare `xargs` per passare gli argomenti a `grep` e facilitare l'esecuzione.

Quindi esattamente come prima cerchiamo "diamond" questa volta, intercettiamo il file, copiamolo nello scrigno e superiamo la missione.

### ESECUZIONE

```
~/Garden/Maze
[mission 41] $ find . -type f | xargs grep -l "diamond"
./76b6faf1a26227e8d41b0d6805530b0/4f99fe47ee0eef54/9c9b22f9

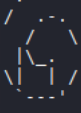
~/Garden/Maze
[mission 41] $ cd 76b6faf1a26227e8d41b0d6805530b0/4f99fe47ee0eef54/

~/Garden/Maze/76b6faf1a26227e8d41b0d6805530b0/4f99fe47ee0eef54
[mission 41] $ mv 9c9b22f9 ~/Forest/Hut/Chest/

~/Garden/Maze/76b6faf1a26227e8d41b0d6805530b0/4f99fe47ee0eef54
[mission 41] $ gc

Congratulations, mission 41 has been successfully completed!
```

## # MISSION 42



**Mission goal**  
=====

Next to the castle, there is a merchant stall. People often buy on credit and reimburse their debt when they can.  
The shopkeeper keeps books on everyone's debt on a scroll. Whenever someone pays his debt, he inscribes "PAID" next to the corresponding transaction.

Combine several commands with ``|`` in order to find the King's debt.

**Remark**  
-----

You are only allowed 3 commands to find the King's debt.  
You can always reset the counter with `gsh reset`, but the whole stall and the debts of everyone will be re-generated as well.

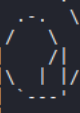
**Hint**  
----

When there are no sub-directories, an alternative to ``find . -name '\*boring\_object\*'`` is to use `ls` and filter the result with grep:  
\$ ls | grep "boring\_object"

**Useful commands**  
=====

grep [OPTIONS] STRING FILE1 ... FILEn  
Filter the files lines, keeping only those that contain the given string.  
If no file is given, `grep` uses stdin.

**Useful option**  
-v : only show the lines that **do not contain** the string.



In questa missione ci viene richiesto di cercare all'interno della directory con tutti i registri di pagamento di un mercante, quanti debiti non saldati ha ancora il Re, sapendo che il mercante, quando un debito viene pagato scrive "PAID" di fianco alla corrispondente transazione.

Andiamo quindi ad utilizzare una sequenza di filtri nel comando per leggere solo le transazioni del Re ancora non saldate:

```
ls | xargs grep "King" | grep -v "PAID"
```

Analizziamo:

```
ls | xargs grep "King"
```

Prende tutti i file della cartella e li invia a **grep** che li filtra in base a quelli contenenti la parola "King"

```
| grep -v "PAID"
```

Gli argomenti in uscita del filtro precedente vengono selezionati in base a quelli che non contengono la parola "PAID", tramite la flag **-v**

Visualizzate le righe che ci interessano, superiamo il livello rispondendo con l'ammontare totale dei debiti del Re.

```
(0)
~/Stall
[mission 42] $ ls | xargs grep "King" | grep -v "PAID"
8983e4e9_s_c_r_o_l_l_8983e4e9769ded98:the King bought a horse for 6 coppers.
8983e4e9_s_c_r_o_l_l_8983e4e9769ded98:the King bought a leather ball for 3 coppers.
8983e4e9_s_c_r_o_l_l_8983e4e9769ded98:the King bought a cow for 6 coppers.
8983e4e9_s_c_r_o_l_l_8983e4e9769ded98:the King bought a pick for 4 coppers.
8983e4e9_s_c_r_o_l_l_8983e4e9769ded98:the King bought a spade for 2 coppers.
8983e4e9_s_c_r_o_l_l_8983e4e9769ded98:the King bought a pick for 4 coppers.
(1)
~/Stall
[mission 42] $ gc
How much does the king owe? 25
Congratulations, mission 42 has been successfully completed!
```

## # MISSION 43

```
Mission goal
=====

Combine several commands with ``|`` in order to find the number of unpaid items.

Remark
-----

You are only allowed a single command.

You can always reset the counter with `gsh reset`, but the whole stall will be re-generated.

Useful commands
=====

grep [OPTIONS] STRING FILE1 ... FILEn
  Filter the files lines, keeping only those that contain the given string.
  If no file is given, the command uses stdin.

  Useful option
    -v : only show the lines that **do not contain** the string.

wc FILE
  Count the number of lines / words / characters in a file
  If no file is given, ``wc`` counts lines / words / characters on stdin.
```

Questa missione ci chiede di scoprire quanti conti non pagati ha il mercate, e di restituire il risultato tutto in unico comando.

Viene introdotto il comando `wc`, che restituisce il numero di linee/parole/caratteri contenute in un file o in qualsiasi input che gli diamo

Analizzando il manuale di `wc`, attraverso il comando `man wc`, possiamo notare che, con la flag `-l` il comando restituisce solo il numero di linee dell'input fornito.

```
-l, --lines
      print the newline counts
```

A questo punto allora, concateniamo i comandi che ci servono, compreso il check del livello e superiamo la missione.

```
(0)
~/Stall
[mission 43] $ ls | xargs grep -v "PAID" | wc -l | gc
How many unpaid items are there?
Congratulations, mission 43 has been successfully completed!
```

## # MISSION 44

```
{ \
Mission goal
=====

A secret message has been found, it is kept in the drawer in Merlin's office. It was probably
enciphered using a Caesar shift cipher.

Decrypt it by making an exhaustive search from the command line.

Hint
----

All other secret messages that have been found were using a shift between 10 and 16.

Useful commands
=====

tr STRING1 STRING2
Replace each character STRING1[i] by STRING2[i] on the standard input, and output the result.
Remark: `tr` is an abbreviation for "translate".

Example: if
  STRING1 = "abcdef"
  STRING2 = "klmnop"
the file will be output with the following substitution (other characters are left unchanged)
a -> k b -> l c -> m
d -> n e -> o f -> p

Note: instead of "abcdefg", it is possible to write "a-g".

This command is particularly useful with a redirection `<`.
/ }
```

In questa missione ci viene chiesto di decriptare un messaggio criptato con un cifrario di Cesare.

Un cifrario di Cesare è un metodo di crittografia che consiste nel sostituire ciascuna lettera di un testo con un'altra lettera a un numero fisso di posizioni più avanti nell'alfabeto.

Ci viene dato un piccolo aiuto, gli shift della cifratura sono da 10 a 16.

Per tradurre questo file viene introdotto il comando:

```
tr STRING1 STRING2
```

Questo comando sostituisce i caratteri della STRING1 con quelli della STRING2.

Lo utilizzeremo sul messaggio criptato, traducendo l'alfabeto a-z con le possibili combinazioni dello stesso shiftato da 10 a 16 caratteri.

Per fortuna sono bastati solo 2 tentativi, in quanto lo shift era di 12 caratteri.

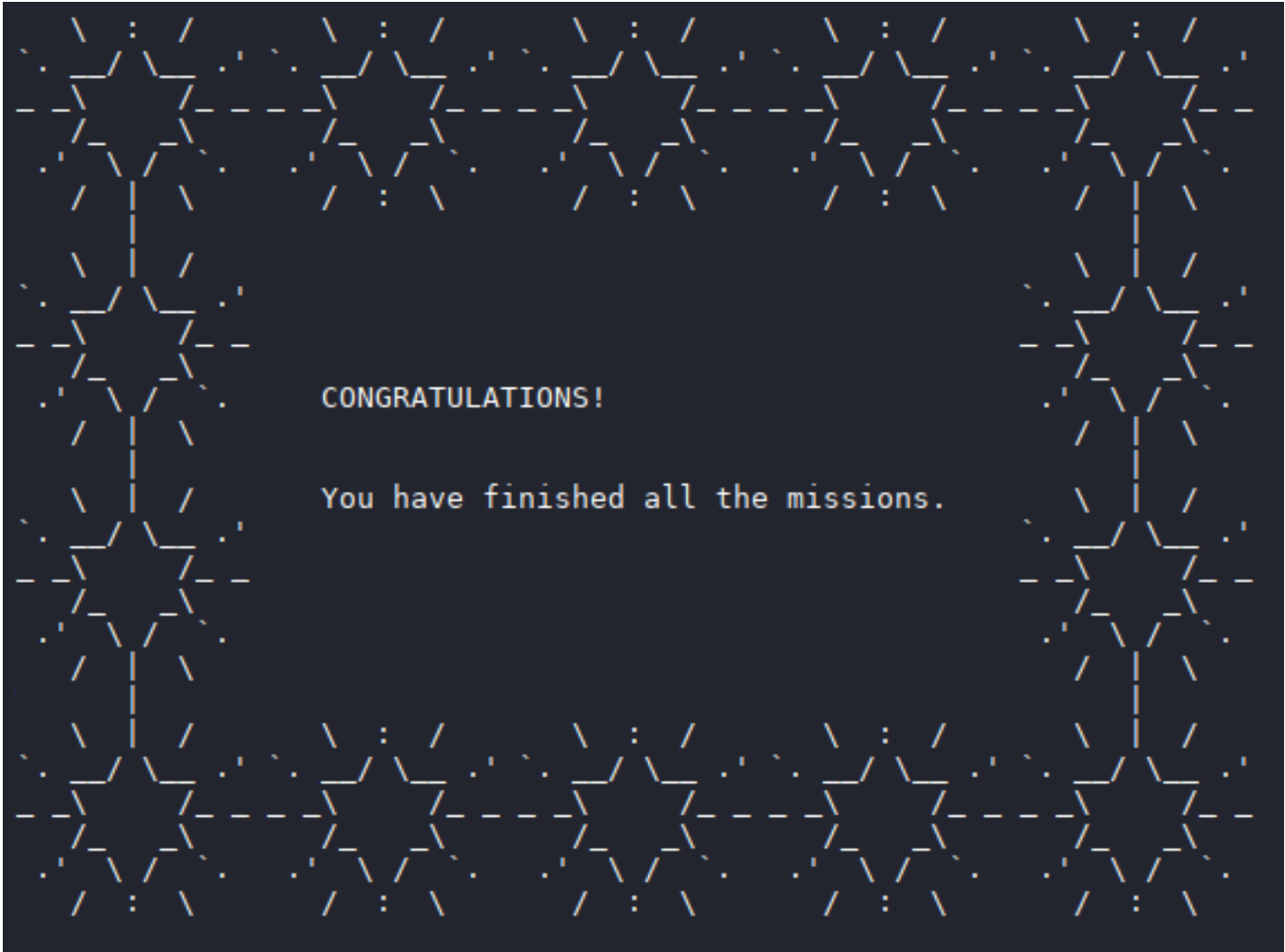
```
~/Castle/Main_building/Library/Merlin_s_office/Drawer
[mission 44] $ cat secret_message | tr "a-z" "mnopqrstuvwxyzabcdefghijkl" > message.txt

~/Castle/Main_building/Library/Merlin_s_office/Drawer
[mission 44] $ cat message.txt
here is my will:
you will get my chest, and everything it contains.
this chest is in the cellar, and the word to make
it re-appear is: qxml
merlin the enchanter
```

Ora che sappiamo la parola d'ordine inseriamola nel check di fine livello:

```
~/Castle/Cellar
[mission 44] $ gc
What's the key that will make Merlin's chest to appear?
qxml

Congratulations, mission 44 has been successfully completed!
```



FINE