Esercizio guidato / 1

Settimana 10
05/12/2022

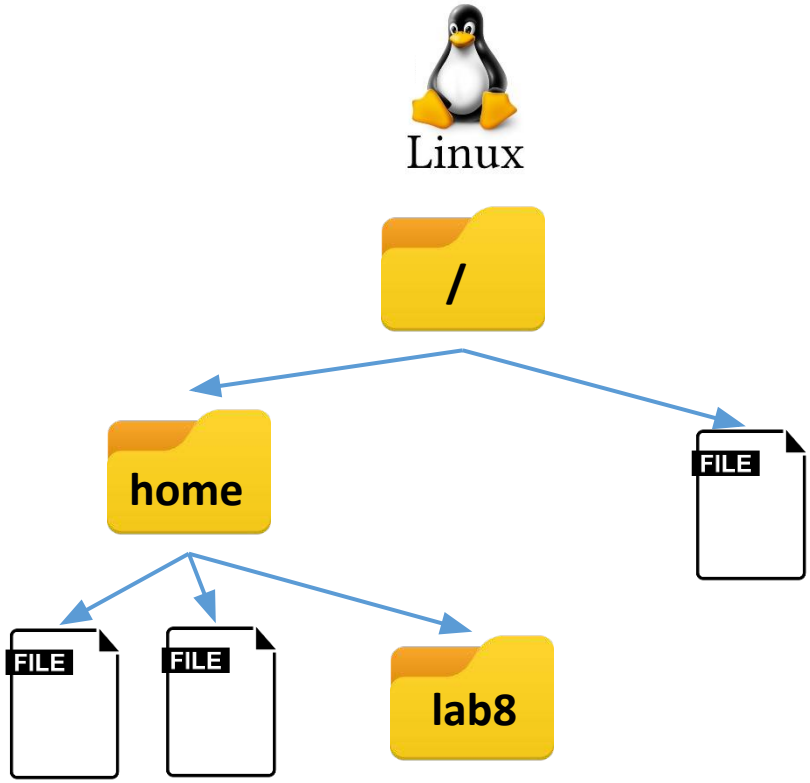
Si ringrazia il Dott. Giacomo Baruzzo per il materiale

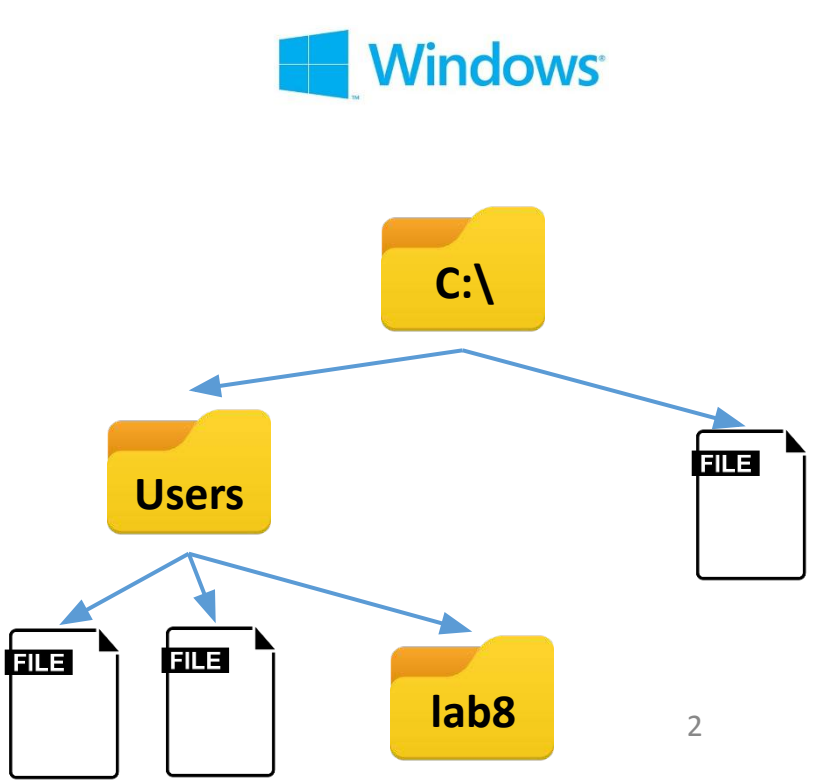
File system

Il *File System* è il sistema con cui sono organizzati file e cartelle nel sistema operativo.

Ha una struttura "gerarchica" (solitamente un albero):

- Ogni cartella ha una cartella padre
- Ogni file è contenuto in una cartella

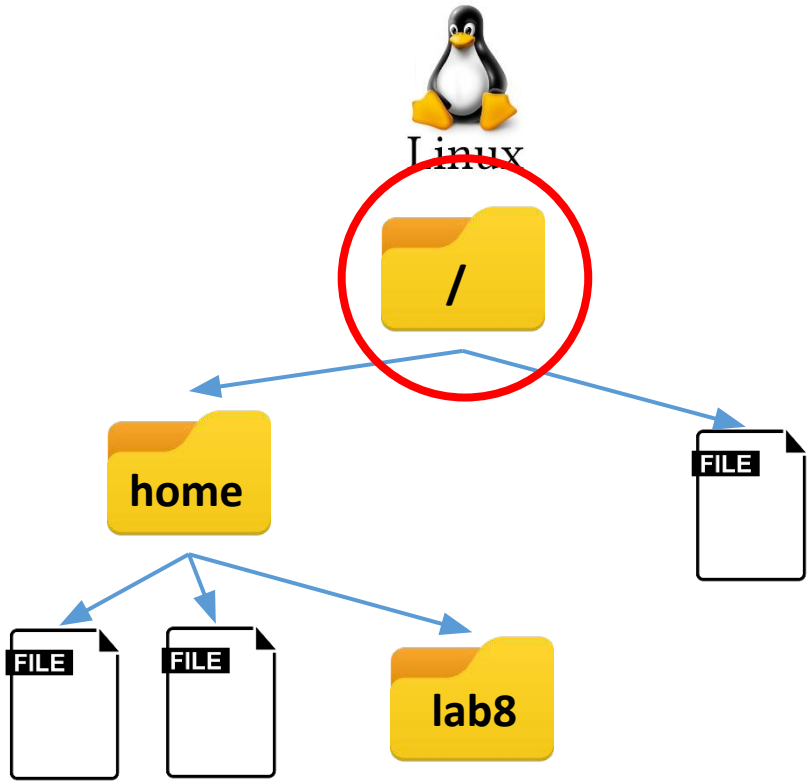


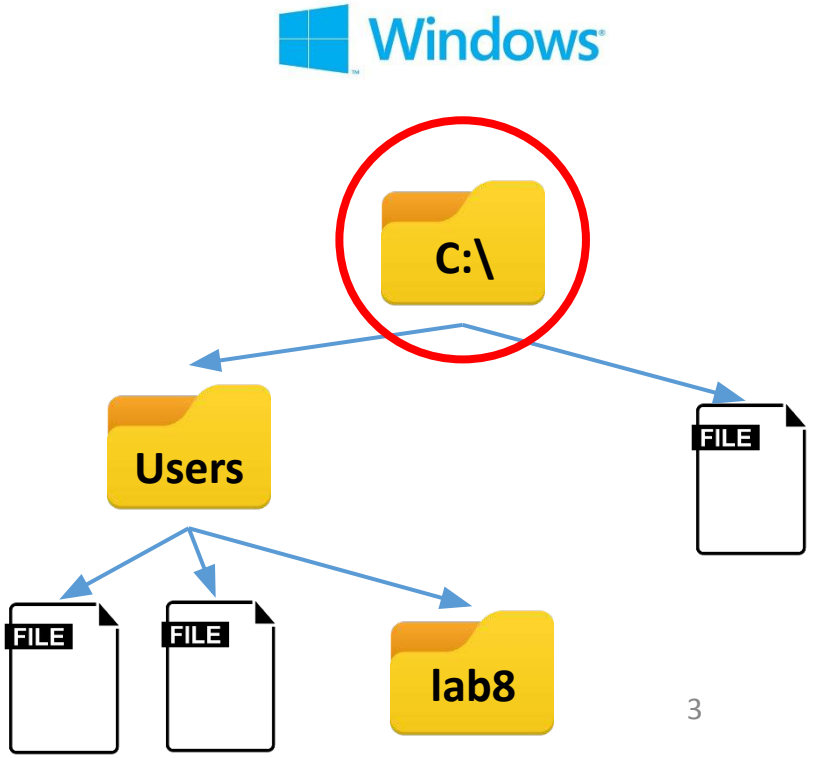


File system: radice

Ogni cartella ha una cartella padre?
No… esiste la "radice"

In sistemi Unix/Linux è "/"
In sistemi Windows è "\", preceduto da il nome del drive (esempio: "C:\")



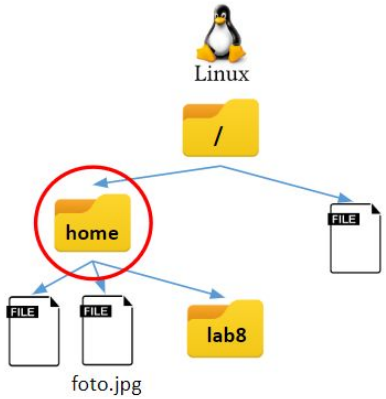


File system: percorsi

La posizione di file e cartelle in un file system viene specificata attraverso i percorsi (o path).
Un percorso può essere **assoluto** o **relativo**.

1) **Percorso assoluto:** parte sempre dalla radice

Esempi:




```
/home/lab8/
/home/foto.jpg
```

```
C:\Users\lab8\
C:\Users\foto.jpg
```

I nomi di file e cartelle sono separati dal carattere separatore:

- "/" (slash) in sistemi Unix/Linux
- "\" (backslash) in sistemi Windows

File system: percorsi

2) **Percorso relativo:** parte sempre dalla cartella "corrente"

Esempio: la cartella corrente è la cartella "home"

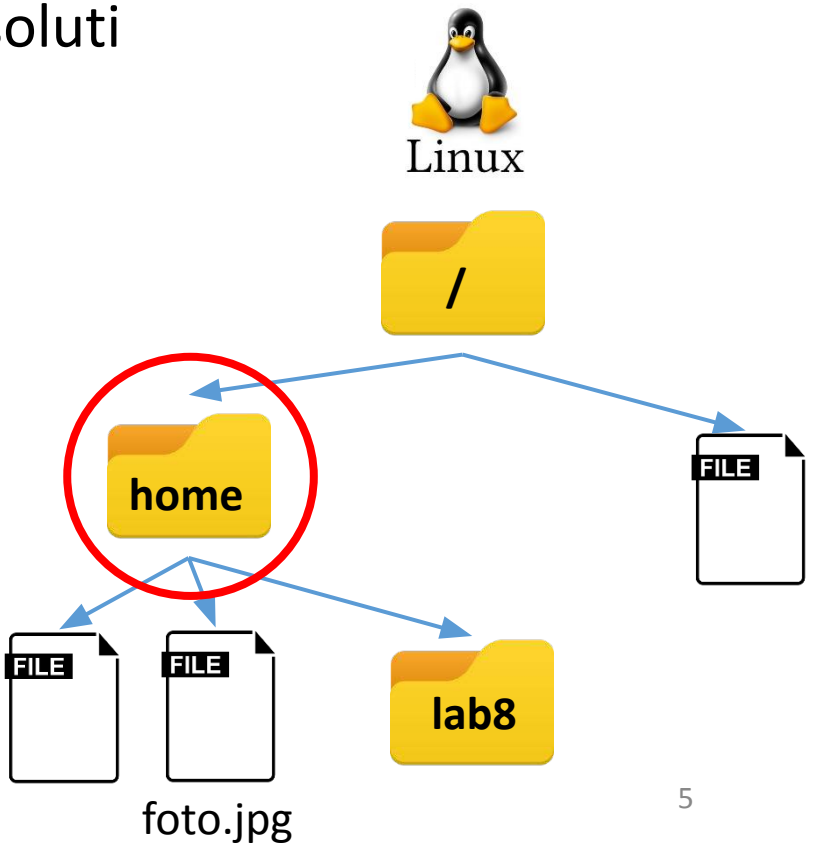
Come posso esprimere i due path assoluti

1. `/home/lab8/`
2. `/home/foto.jpg`

in forma di path relativi?

```
./lab8/
./foto.jpg
```

La cartella corrente si indica con il carattere "."



File system: percorsi

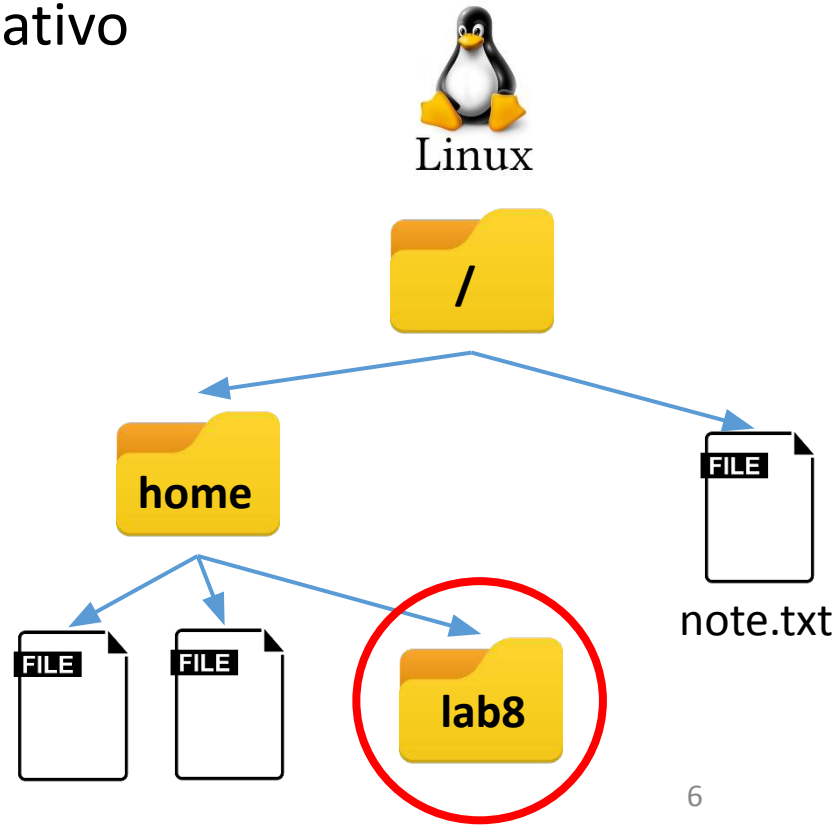
2) **Percorso relativo:** parte sempre dalla cartella "corrente"

Esempio: la **cartella corrente** è la cartella "lab8"

Come posso esprimere in un path relativo
il percorso fino al file note.txt ?

```
../../note.txt
```

Per indicare la cartella padre si usa la notazione ".."



Interazione Python – File system

- Python può interagire con il file system
  - Creare/rimuovere una cartella
  - Rimuovere un file
  - Navigare nel file system
  - …
- Per fare ciò, esistono i moduli `os` e `os.path`

Carattere separatore e percorsi

Variabile che contiene il carattere separatore

`os.sep`

"/" (Unix/Linux) o "\\" (Windows)

Variabile che contiene il path relativo della cartella corrente

`os.curdir`

"."

Variabile che contiene il path relativo della cartella padre

`os.pardir`

".."

Funzione per ottenere il path assoluto della cartella corrente

`os.getcwd()`

Esempio: "/home/lab8/"

File e cartelle

Funzione per ottenere il contenuto di una cartella

```
os.listdir()
```

Ritorna il contenuto della cartella corrente

```
os.listdir("/home/lab8/")
```

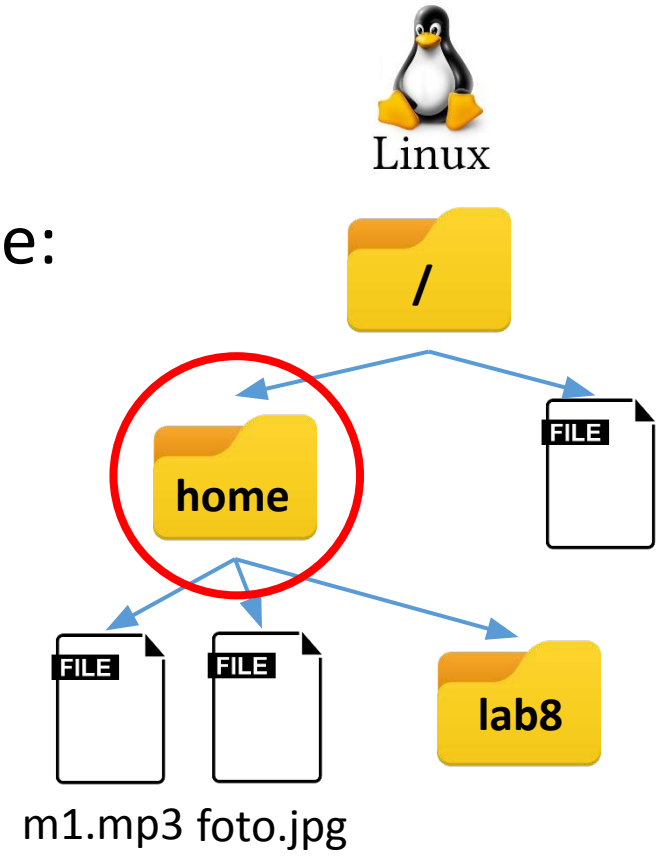
Ritorna il contenuto della cartella "/home/lab8"

File e cartelle: esempi

Cartella corrente: "home"

La funzione `os.listdir()` restituisce:

['m1.mp3', 'foto.jpg', 'lab8']



NOTE:

1. Nessuna indicazione se si tratta di file o cartella
2. Sono indicati solo i nomi (non i path, né "." e "..")

File e cartelle

Verificare se il path indica un file

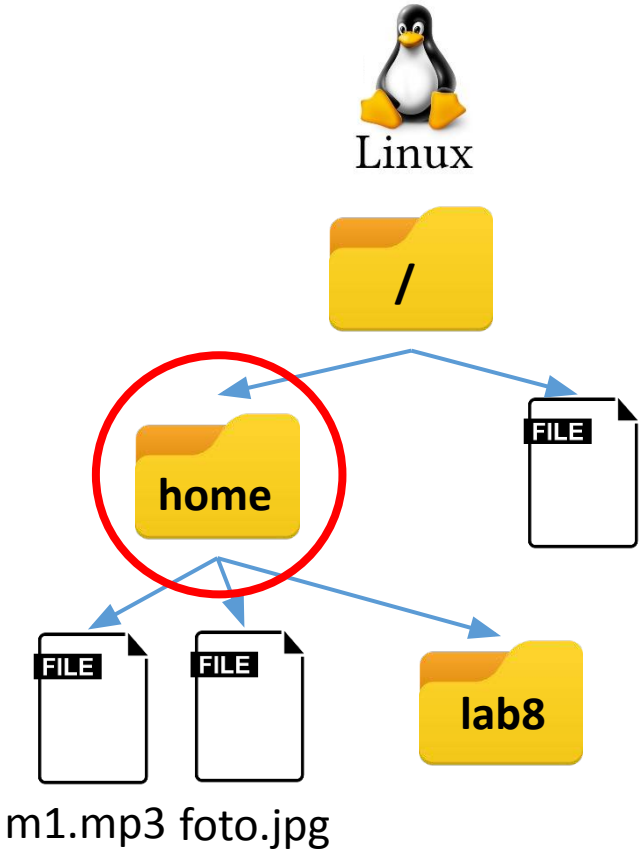
`os.path.isfile(path)`

Verificare se il path indica un cartella

`os.path.isdir(path)`

Esempi:

```
os.path.isfile("/home/lab8/") False
os.path.isdir("/home/lab8/") True
os.path.isfile("/home/foto.jpg") True
os.path.isdir("/home/foto.jpg") False
os.path.isdir("/home/lab9/") False
```



Dimensione di file

Funzione per ottenere la dimensione di un file (in byte)

```
os.path.getsize(path)
```

Esempio:

```
os.path.getsize("/home/foto.jpg")
```

Ritorna

```
7985
```

File system e ricorsione

Andiamo vedere un paio di esempi di applicazione della **interazione tra Python e il file system** usando:

- i **moduli/funzioni/variabili appena introdotti**…
- il costrutto delle **ricorsione**

RICHIAMO: La ricorsione

Per risolvere un problema con la ricorsione, bisogna scomporlo in 2 parti:

- **CASO BASE:** parte del problema risolvibile senza dover ricorrere alla ricorsione. Corrisponde solitamente ad un caso "semplice" del problema in esame, per il quale è possibile specificare una soluzione esplicita (non ricorsiva)
- **CASO RICORSIVO:** parte del problema che rappresenta un problema omogeneo a quello iniziale, ma di "taglia" più piccola

Esempio di applicazione 1

Scrivere un programma Python **displayNames.py** che navighi in modo ricorsivo il file system a partire da una directory assegnata.

Per navigazione ricorsiva si intende che, per ogni cartella presente nella directory assegnata, il programma deve:

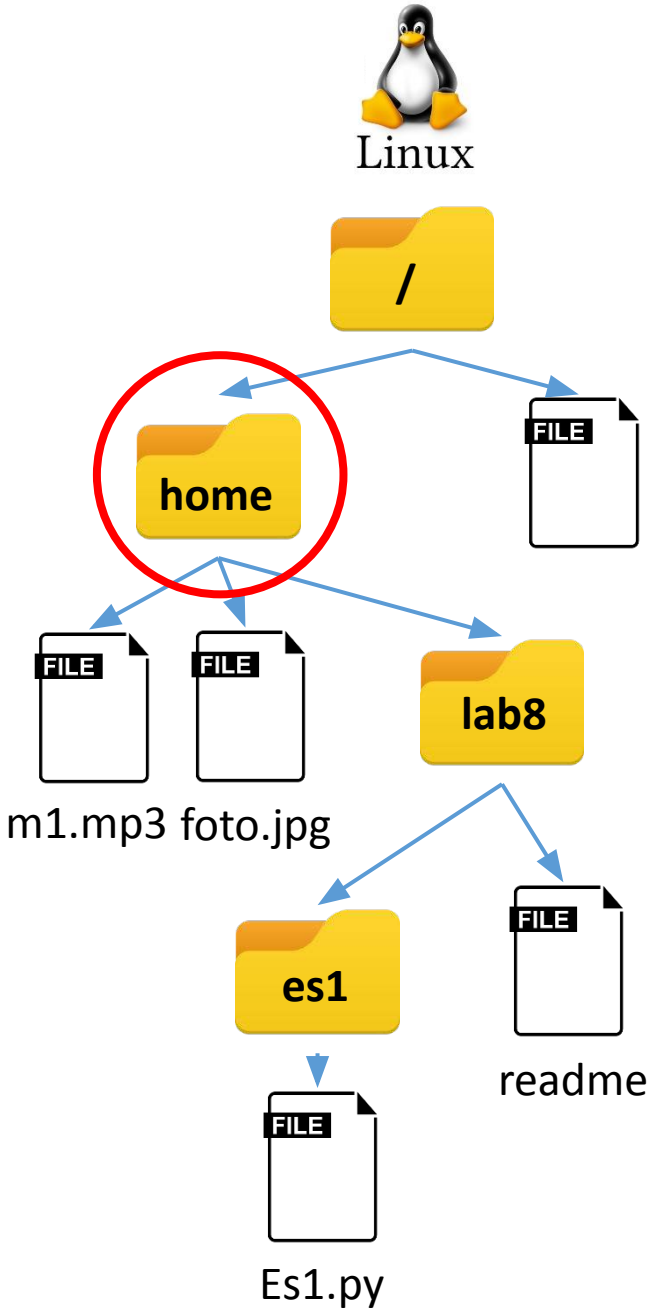
1. visualizzare il nome completo di tutti i file
2. visualizzare il nome completo di tutte le cartelle
3. "entrare" in tutte le cartelle e ripetere i punti 1), 2) e 3)

Il nome dei file andrà preceduto dal carattere "F", il nome delle cartelle andrà preceduto dal carattere "D".

Esempio

```
displayNames("/home/")

F /home/m1.mp3
F /home/foto.jpg
D /home/lab8
D /home/lab8/es1
F /home/lab8/es1/Es1.py
F /home/lab8/readme
```



Codice Python

```
import os
import os.path

def displayNames(dir) :
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return
  else :
    for name in names :
      if os.path.isfile(dir + os.sep + name) :
        print("F ", end="")
        print(dir + os.sep + name)
      else :
        print("D ", end="")
        print(dir + os.sep + name)
        displayNames(dir + os.sep + name)

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
displayNames(dir)
```

Codice Python

```
import os
import os.path

def displayNames(dir) :
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return
  else :
    for name in names :
      if os.path.isfile(dir + os.sep + name) :
        print("F ", end="")
        print(dir + os.sep + name)
      else :
        print("D ", end="")
        print(dir + os.sep + name)
        displayNames(dir + os.sep + name)

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
displayNames(dir)
```

Codice Python

```
import os
import os.path

def displayNames(dir) :
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return
  else :
    for name in names :
      if os.path.isfile(dir + os.sep + name) :
        print("F ", end="")
        print(dir + os.sep + name)
      else :
        print("D ", end="")
        print(dir + os.sep + name)
        displayNames(dir + os.sep + name)

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
displayNames(dir)
```

Codice Python

```
import os
import os.path

def displayNames(dir) :
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return
  else :
    for name in names :
      if os.path.isfile(dir + os.sep + name) :
        print("F ", end="")
        print(dir + os.sep + name)
      else :
        print("D ", end="")
        print(dir + os.sep + name)
        displayNames(dir + os.sep + name)

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
displayNames(dir)
```

Listo tutti gli elementi della cartella corrente

Gestisco il caso in cui il path della cartella termini con il carattere separatore, rimuovendolo

Codice Python

```
import os
import os.path

def displayNames(dir) :
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return
  else :
    for name in names :
      if os.path.isfile(dir + os.sep + name) :
        print("F ", end="")
        print(dir + os.sep + name)
      else :
        print("D ", end="")
        print(dir + os.sep + name)
        displayNames(dir + os.sep + name)

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
displayNames(dir)
```

Caso base 1: la cartella è vuota

Codice Python

```
import os
import os.path

def displayNames(dir) :
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return
  else :
    for name in names :
      if os.path.isfile(dir + os.sep + name) :
        print("F ", end="")
        print(dir + os.sep + name)
      else :
        print("D ", end="")
        print(dir + os.sep + name)
        displayNames(dir + os.sep + name)

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
displayNames(dir)
```

Se non è vuota, inizio a scorrere gli elementi

Codice Python

```
import os
import os.path

def displayNames(dir) :
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return
  else :
    for name in names :
      if os.path.isfile(dir + os.sep + name) :
        print("F ", end="")
        print(dir + os.sep + name)
      else :
        print("D ", end="")
        print(dir + os.sep + name)
        displayNames(dir + os.sep + name)

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
displayNames(dir)
```

Caso base 2: sto analizzando un file

Codice Python

```
import os
import os.path

def displayNames(dir) :
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return
  else :
    for name in names :
      if os.path.isfile(dir + os.sep + name) :
        print("F ", end="")
        print(dir + os.sep + name)
      else :
        print("D ", end="")
        print(dir + os.sep + name)
        displayNames(dir + os.sep + name)

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
displayNames(dir)
```

Caso ricorsivo: sto analizzando una cartella

Esempio di applicazione 2

Scrivere un programma **dirSize.py** che calcola la somma dello spazio occupato dai file di una cartella e ricorsivamente per tutte le sottocartelle (usando `os.path.getsize()`).

Analizzando una cartella, si fa la somma delle dimensioni ricevute dalle invocazioni ricorsive sulle sottocartelle, poi si aggiungono le dimensioni dei file presenti nella cartella e si restituisce il totale.

Il problema è simile a quello di prima… La differenza sta nel fatto che bisogna ottenere la dimensione dei file (anziché stampare il loro nome) e tenere traccia della somma delle dimensioni.

Codice Python

```
import os
import os.path

def dirSize(dir) :
  size_val = 0
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return 0
  else :
    for name in names :
      elem_path = dir + os.sep + name
      if os.path.isfile(elem_path) :
        size_val += os.path.getsize(elem_path)
      else :
        size_val += dirSize(elem_path)
  return size_val

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
dir_size = dirSize(dir)
print(dir_size)
```

Codice Python

```
import os
import os.path

def dirSize(dir) :
  size_val = 0
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return 0
  else :
    for name in names :
      elem_path = dir + os.sep + name
      if os.path.isfile(elem_path) :
        size_val += os.path.getsize(elem_path)
      else :
        size_val += dirSize(elem_path)
  return size_val

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
dir_size = dirSize(dir)
print(dir_size)
```

Inizializzo una variabile per tenere traccia della dimensione cumulativa dei file

Codice Python

```
import os
import os.path

def dirSize(dir) :
  size_val = 0
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return 0
  else :
    for name in names :
      elem_path = dir + os.sep + name
      if os.path.isfile(elem_path) :
        size_val += os.path.getsize(elem_path)
      else :
        size_val += dirSize(elem_path)
  return size_val

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
dir_size = dirSize(dir)
print(dir_size)
```

Caso base 1: cartella vuota

Codice Python

```
import os
import os.path

def dirSize(dir) :
  size_val = 0
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return 0
  else :
    for name in names :
      elem_path = dir + os.sep + name
      if os.path.isfile(elem_path) :
        size_val += os.path.getsize(elem_path)
      else :
        size_val += dirSize(elem_path)
  return size_val

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
dir_size = dirSize(dir)
print(dir_size)
```

Caso base 2: sto analizzando un file

Codice Python

```
import os
import os.path

def dirSize(dir) :
  size_val = 0
  names = os.listdir(dir)
  if dir.endswith(os.sep) :
    dir = dir[:-1]
  if len(names) == 0 :
    return 0
  else :
    for name in names :
      elem_path = dir + os.sep + name
      if os.path.isfile(elem_path) :
        size_val += os.path.getsize(elem_path)
      else :
        size_val += dirSize(elem_path)
  return size_val

dir = input("Partiamo da: ")
if len(dir) == 0 :
  dir = os.getcwd()
dir_size = dirSize(dir)
print(dir_size)
```

Caso ricorsivo: sto analizzando una cartella