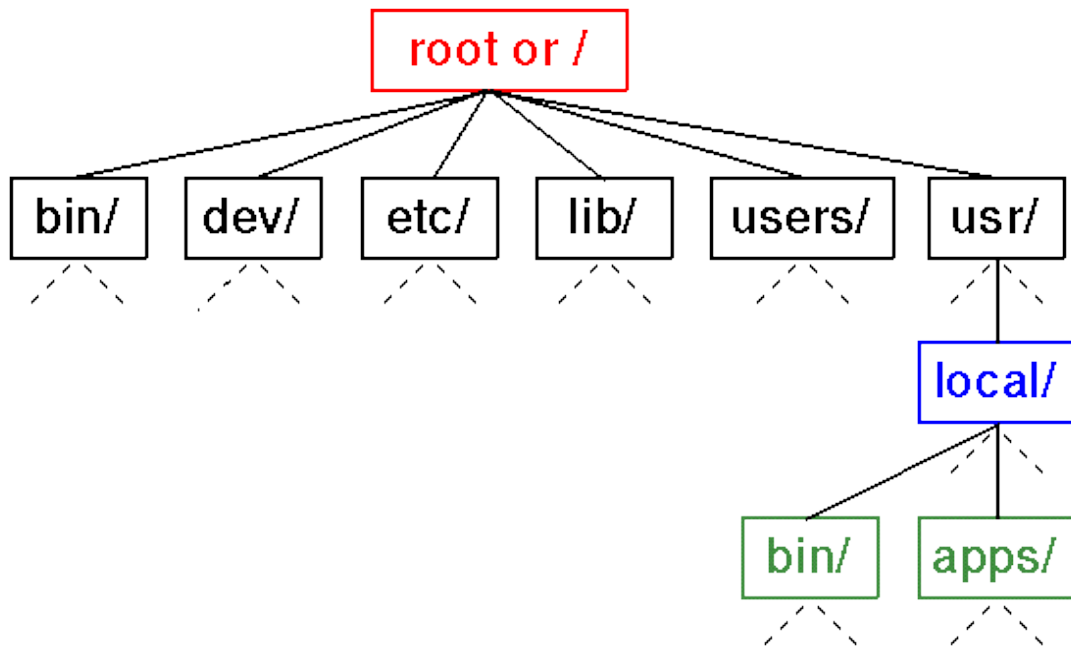


Lezione 10

- La cartella Lezione10 contiene un file cartesiano.py con un'implementazione completa del metodo overlaps().

Modello di *file system*.

Questo esercizio è di fondamentale importanza ai fini del ripasso e della comprensione di tutti gli argomenti di programmazione python. La complessità è elevata ma consente un'ottima preparazione per la provetta e per gli esami.



Si progetti un modello di *file system* in cui sia possibile stampare la cartella di lavoro corrente, cambiare cartella di lavoro, elencare file e sotto cartelle di una cartella, ottenere il percorso di un file, cercare un file all'interno dell'albero delle cartelle, stampare l'albero dei file e delle sotto cartelle a partire dalla cartella corrente.

Classi

File

Avrà due metodi: `getName`, per restituire il nome del file e `isDir`, che restituisce sempre `False`.

Costruttore: accetta una stringa con il nome del file e un altro oggetto di tipo `Dir` come genitore. Nel costruttore, aggiunge se stesso come figlio alla `directory` che viene passata come argomento, controllando che essa sia una `directory` tramite il metodo `isDir()` di `Dir` e che essa sia diversa da `None`.

Dir

È un **File**, a cui è possibile aggiungere altri file o `directory` figli tramite `addChild` e da cui è possibile ottenere la lista dei **File** figli tramite `getChildren`. Reimplementa `isDir` restituendo `True`.

Costruttore: equivalente a quello di `File`.

FileManager

A questa classe sono associati N file e directory, organizzati ad albero, come mostrato in figura.

Operazioni:

- `pwd()`: *print working directory*, stampa il nome della cartella in cui ci si trova in quel momento;
- `ls()`: *list directory*: elenca file e directory all'interno della cartella in cui mi trovo;
- `cd(nomeDir: String)`: cambia la directory corrente in quella specificata con `nomeDir`, che deve esistere ed essere figlia diretta della cartella corrente.
- `path(file: File)`: restituisce una stringa con il percorso completo del file passato in ingresso, ad esempio: `/home/giacomo/universita/2012/provetta3.py`;
- `find(nomeFile: stringa)`: trova un file con il nome dato all'interno dell'albero, in modo ricorsivo...
- `tree()`: stampa l'albero delle directory e dei file a partire dalla directory corrente, ad esempio:

Costruttore: accetta come parametro la root directory ("/").

```
>>> f.tree()
+ /
+ boot
+ etc
+ conf.d
  - rc.conf
  - fstab
+ usr
+ bin
  - gcc
  - vi
+ local
+ bin
  - bash
  - python
+ home
+ giacomo
+ universita
+ 2011
+ 2012
  - biblioteca.py
  - provetta3.py
+ anna
+ images
  - anna1.jpg
  - anna2.jpg
+ holidays
  - holidays1.jpg
  - holidays2.jpg
  - holidays3.jpg
  - holidays4.jpg
  - holidays5.jpg
```

Svolgimento

Si scriva il diagramma UML delle tre classi necessarie.

Si implementi in python la definizione delle classi.

Si crei la struttura di directory mostrata alla pagina precedente.

Si costruisca infine un oggetto FileManager con la directory root, che e` la radice dell'albero, ed e` unica (tutto il resto dell'albero e` figlio della root directory, come nell'immagine sopra).

Si usi il modello per cambiare directory, trovare file e stamparne il percorso tramite find() e path(), si elenchi il contenuto di directory e si usi tree() per disegnare l'albero sottostante la directory corrente. pwd() stampa la directory corrente.

Soluzioni.

Files:

- *filesystem_defs.py*: contiene le definizioni delle classi;
- *filesystem.py*: alloca la struttura della directory mostrata nella pagina precedente;
- *filesystem.txt*: esempi d'uso del modello da console python, commentato.

Note.

Si segnalino eventuali problemi riscontrati nell'esercizio.

Si tratta di un esercizio preparato per un orale dell'anno 2010-2011.