os.walk() が使える。
普通の言語のように、起点ディレクトリを表すオブジェクトを取得（または作成）して、そのメソッドやプロパティを参照する必要はない。

```
import os
for dir_path, subdir_name_list, file_name_list in os.walk(top_dir_path):
    ...
```

dir_path: walk 対象となったディレクトリのパス。
subdir_name_list: dir_path 内に存在するサブディレクトリ名のリスト。
file_name_list: dir_path 内に存在するファイル名のリスト。

各リスト内の項目は、あらかじめ昇順にソートされているようだ。

os.walk() は、(dir_path, subdir_name_list, file_list) というタプルを返すイテレータを生成する。
タプルは os.walk() の引き数のディレクトリパスを起点とし、以降、subdir_name_list のサブディレクトリ・リストに対してイテレートされていく。

特定のディレクトリをサーチパスからはずすには、イテレータの次のエントリが呼ばれる前に、subdir_name_list 内からサブディレクトリ名を削除する（ただし、以下の topdown オプションが True のときに限られる）。

os.walk() には名前つき引き数でオプションを指定できる。
topdown オプション引き数は、ディレクトリ walk の方向を指定する。

```
for dir, subdirs, files in os.walk(top_dir, topdown=False):
    print dir, subdirs, files
```

"topdown=False" はディレクトリツリーをボトムアップで walk する。
topdown 引き数を指定しなければ、"topdown=True" と同じ意味。

onerror オプション引き数に指定する値は、OSError オブジェクトひとつを引き数として取る関数。

```
import os
def error_handler(e):
    print e
for dir, subdirs, files in os.walk(top_dir, onerror=error_handler):
   ...
```

ディレクトリのリスティングでエラーが発生したとき、onerror 引き数で指定した関数に OSError オブジェクトのインスタンスが渡され、呼び出される。
"onerror=None" だと、ディレクトリの walking 中にエラーが発生しても無視される（これがデフォルト）。

topdown と onerror は同時に指定できる。