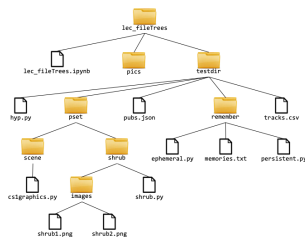


# Recursive File Tree Traversal

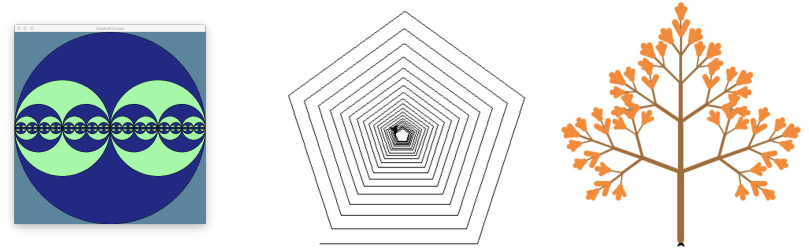


CS111 Computer Programming

Department of Computer Science  
Wellesley College

## Motivation: Practical Recursion on File Systems

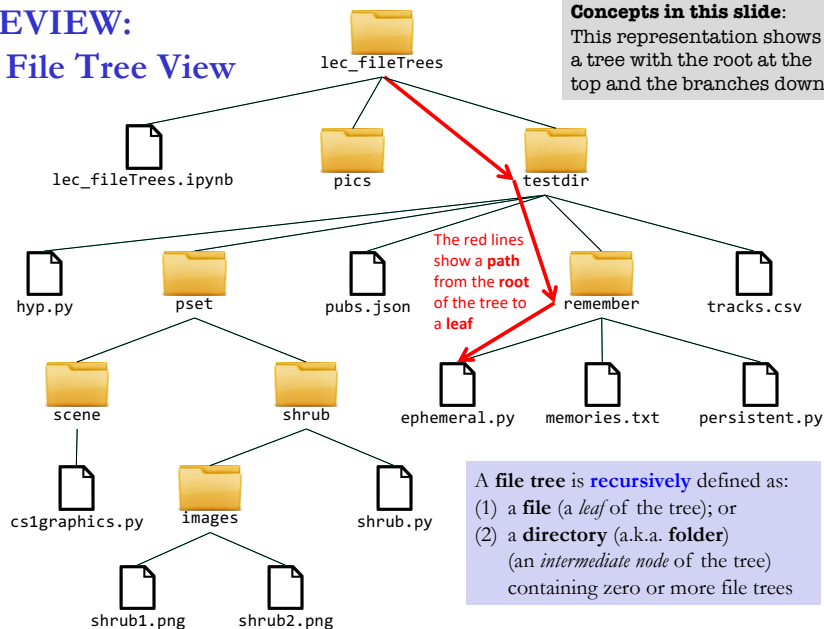
So far we have used recursion to make self-similar pictures with Turtles.



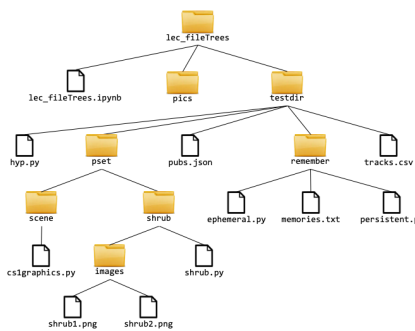
But can recursion be used to solve more practical problems? **YES!**

Today we'll see that the file system on your computer is organized in a recursive way, and so it's most natural to process it using recursive functions. There are many practical recursive functions that "walk" over trees of folders and files.

## REVIEW: A File Tree View



## File Tree Traversal: Print all Dirs and Files



```
In [25]: printFileTree('testdir')
testdir
testdir/hyp.py
testdir/pset
testdir/pset/scene
testdir/pset/scene/cs1graphics.py
testdir/pset/shrub
testdir/pset/shrub/images
testdir/pset/shrub/images/shrub1.png
testdir/pset/shrub/images/shrub2.png
testdir/pset/shrub/shrub.py
testdir/pubs.json
testdir/remember
testdir/remember/ephemeral.py
testdir/remember/memories.txt
testdir/remember/persistent.py
testdir/tracks.csv
```

**printFileTree** is a recursive function that visits all nodes of the file tree and prints out the relative pathnames of all the nonhidden files and directories it encounters.

```
In [26]: printFileTree('testdir/hyp.py')
testdir/hyp.py

In [27]: printFileTree('testdir/remember')
testdir/remember
testdir/remember/ephemeral.py
testdir/remember/memories.txt
testdir/remember/persistent.py
```

## File Tree Traversal: Broken Version

```
def printFileTreeBroken(root):  
    '''Print all directories and files  
    (one per line) starting at root,  
    which is a directory or file name'''  
    if os.path.isfile(root):  
        print(root)  
    elif os.path.isdir(root):  
        print(root)  
        for fileOrDir in os.listdir(root):  
            printFileTreeBroken(fileOrDir)
```

Implicit `else: pass` at end handles other file types and nonexistent file names by doing nothing for them.

```
In [28]: printFileTreeBroken('testdir')  
testdir  
.DS_Store
```

Why does this print only two names?

```
In [29]: os.listdir('testdir')  
Out[29]: ['.DS_Store', '.numbers', 'hyp.py', 'pset', 'pubs.json',  
'remember', 'tracks.csv']
```

```
In [30]: os.path.isfile('hyp.py')  
Out[30]: False
```

```
In [31]: os.path.isfile('testdir/hyp.py')  
Out[31]: True
```

Recursive File Tree Traversal 5

## File Tree Traversal: Better Version

**Concepts in this slide:**  
Fixes the error from slide 5, by providing the complete relative pathname.

```
def printFileTreeBetter(root):  
    if os.path.isfile(root):  
        print(root)  
    elif os.path.isdir(root):  
        print(root)  
        for fileOrDir in os.listdir(root):  
            printFileTreeBetter(os.path.join(root, fileOrDir))
```

This acts like `root + '/' + fileOrDir`, but is clearer and less error prone.

```
In [32]: printFileTreeBetter('testdir')  
testdir  
testdir/.DS_Store  
testdir/.numbers  
testdir/hyp.py  
testdir/pset  
testdir/pset/.DS_Store  
testdir/pset/scene  
testdir/pset/scene/cslgraphics.py  
...
```

Dot files are still printed, but we'd like to hide them.

Recursive File Tree Traversal 6

## Testing for Hidden Files

```
def isHiddenFile(path):  
    base = os.path.basename(path)  
    return (len(base) > 0  
            and base[0] == '.'  
            and base != '..')  
            and base != '..')
```

'.' (current directory) and '..' (parent directory) are special cases of paths beginning with dots that are not hidden.

```
In [32]: isHiddenFile('.DS_Store')  
Out[32]: True
```

```
In [33]: isHiddenFile('testdir/psets/.DS_Store')  
Out[33]: True
```

```
In [35]: isHiddenFile('testdir/psets/memories.txt')  
Out[35]: False
```

```
In [36]: isHiddenFile('.')  
Out[36]: False
```

Recursive File Tree Traversal 7

## File Tree Traversal: Final Correct Version that Filters out Dot Files

```
def printFileTree(root):  
    if isHiddenFile(root):  
        pass # filter out dot files  
    elif os.path.isfile(root):  
        print(root)  
    elif os.path.isdir(root):  
        print(root)  
        for fileOrDir in os.listdir(root):  
            printFileTree(os.path.join(root, fileOrDir))
```

```
In [32]: printDirectoryTree('testdir')  
testdir  
testdir/hyp.py  
testdir/pset  
testdir/pset/scene  
testdir/pset/scene/cslgraphics.py  
testdir/pset/shrub  
testdir/pset/shrub/images  
testdir/pset/shrub/images/shrub1.png  
testdir/pset/shrub/images/shrub2.png  
...
```

Recursive File Tree Traversal 8

## Exercise 1: File Tree Traversal: Print Files Only

```
def printFiles(root):  
    '''Print only the (nondirectory) files encountered  
    in a file tree traversal starting at root'''  
    # flesh this out
```



```
In [35]: printFiles('testdir')  
testdir/hyp.py  
testdir/pset/scene/cslgraphics.py  
testdir/pset/shrub/images/shrub1.png  
testdir/pset/shrub/images/shrub2.png  
testdir/pset/shrub/shrub.py  
testdir/pubs.json  
testdir/remember/ephemeral.py  
testdir/remember/memories.txt  
testdir/remember/persistent.py  
testdir/tracks.csv
```

Recursive File Tree Traversal 9

## Exercise 2: File Tree Traversal: Print Dirs Only

```
def printDirs(root):  
    '''Print only the directories encountered in a  
    file tree traversal starting at root'''  
    # flesh this out
```



```
In [34]: printDirs('testdir')  
testdir  
testdir/pset  
testdir/pset/scene  
testdir/pset/shrub  
testdir/pset/shrub/images  
testdir/remember
```

Recursive File Tree Traversal 10

## Exercise 3: File Tree Traversal: Count Files Only

```
def countFiles(root):  
    '''Returns the number of (nondirectory) files encountered  
    in a file tree traversal starting at root'''  
  
    if isHiddenFile(root):  
        return ?? # What goes here?  
    elif os.path.isfile(root):  
        return ?? # What goes here?  
    elif os.path.isdir(root):  
        # flesh this out
```



```
In [37]: countFiles('testdir')  
Out[37]: 10
```

Recursive File Tree Traversal 11

## Exercise 4: File Tree Traversal: Count Dirs Only

```
def countDirs(root):  
    '''Returns the number of directories encountered in a  
    file tree traversal starting at root'''  
  
    if isHiddenFile(root):  
        return ?? # What goes here?  
    elif os.path.isdir(root):  
        # flesh this out  
  
    else:  
        return ?? # What goes here?
```



```
In [36]: countDirs('testdir')  
Out[36]: 6
```

Recursive File Tree Traversal 12