



write a python command to print out the size of a directory in bytes



Certainly! You can use the `os` module in Python to get the size (in bytes) of a directory recursively. Here's a **single command (one-liner)** you can run in your terminal or use in a Python script:

As a Python command (script):

```
import os
size = sum(os.path.getsize(p) for p in os.walk('/PATH/TO/DIRECTORY')) -
sum(os.stat(d).st_size for d in os.listdir('/PATH/TO/DIRECTORY'))
print(size)
```

Explanation:

- `os.walk(...)` lists all files and subdirectories.
- `os.path.getsize(p)` gives the size of each **file** (not including directories themselves).
- `os.stat(d).st_size` gives the **size of a directory entry** (in bytes), but directories themselves don't store this value; this is just the filesystem's metadata.
 - For **true disk usage** of a directory (including subdirs, but not counting the parent), you need to **subtract the size of the parent directory** once (for the root of the walk).
- For a **non-recursive** version (only direct files in the given directory):

Non-recursive (only files directly in the directory):

```
import os

dirpath = '/PATH/TO/DIRECTORY'
total = sum(os.path.getsize(f) for f in os.listdir(dirpath) if
os.path.isfile(os.path.join(dirpath, f)))
print(total)
```

Fully Recursive Size (with proper subtraction for directories):

For a **recursive directory size in bytes**, the most accurate way is to use `shutil.disk_usage`:

```
import shutil

dirpath = '/PATH/TO/DIRECTORY'
size = shutil.disk_usage(dirpath).total_bytes # Includes everything
below (files + dir metadata)
print(size)
```

Note:

`shutil.disk_usage()` counts **all bytes used on the disk for this directory tree**, including:

- File contents
- Directory inodes
- Block allocation
- Metadata
- Possibly more (depending on filesystem and OS).

If you want **only the sum of file sizes** (not counting directory structures or OS overhead):

Sum of all file sizes in directory tree (recursive):

```
import os

def get_file_sizes(path):
    total = 0
    for dirpath, dirnames, filenames in os.walk(path):
        for filename in filenames:
            fpath = os.path.join(dirpath, filename)
            try:
                total += os.path.getsize(fpath)
            except OSError:
                pass # Skip unreadable files
    return total

dirpath = '/PATH/TO/DIRECTORY'
print(get_file_sizes(dirpath))
```

Replace `/PATH/TO/DIRECTORY` with your target path.

Let me know if you want to exclude subdirectory sizes, or only count leaf files, etc.!