File Backup

Written by Youssef Beltagy on 11/14/2020 for CSS436 in AUT2020.

This program uploads a directory to AWS's S3 storage and allows for retrieval. It maintains the relative directory hieraracy.

Build and Run

I wrote this program in python, so there is no reason to build it.

To run the program:

```
# usage: python backup.py <download|upload> <dir_name> <bucket_name>
python backup.py upload tests chicken-backup
```

Or

```
python backup.py download output chicken-backup
```

I use python 3.9 with boto3. I expect that boto3 is configured. I developed and tested this program on a Windows machine. I expect it to work on Linux machines as well, but I have not tested that because I don't have boto3 on my WSL.

Design

To upload, I used recursion. I recur through the directories and upload their files. To ensure I don't lose empty directories, I upload empty stubs that end with "/" to represent directories. "/" is not acceptable in a file name, so this does not introduce bugs. To maintain hierarchal information, I prepend the whole path of a file to its name to make the S3 object key.

Before I upload, I make a set of all keys in the bucket. As I upload, I pop all the keys uploaded from the set. After uploading, the keys in the set are remnants from previous uploads. So I traverse the set and delete the objects with these keys.

When I'm uploading, I check that the file I'm uploading doesn't exist in the AWS bucket or was modified after the version in the AWS bucket was uploaded.

The upload algorithm looks like this:

```
def upload (dir)
    for every file in dir:
        if file is a file, upload it.
        else if file is a directory, call upload(file)
```

To download, I use a for loop. S3 is flat non-hierarchical storage. So the for loop gives me all the objects in the storage. I split the file name from its path and use os.mkdirs() to make those directories. Then I write the file.

When I download, I don't delete what is in the directory. So the directory after downloading will contain all of what it used to contain in addition to what is in the S3 bucket.

When I download, if the file and object share the same name and relative location, then the newer one remains. This wasn't stated clearly in the assignment, so it really could be implemented either way and it will make sense. This means that if you upload from a file, and then download to it multiple times. The first download will overwrite the original files, but subsequent downloads won't be performed.

This could have been done in the opposite way so that the older of the two remains. That is still reasonable behavior (I want to store the old value), but it would mean that downloading consecutively to the same directory will overwrite for no reason because the downloaded files are more recently modified than the S3 bucket.

The assignment wasn't clear, so I developed based on my understanding. Changing between those two modes is as easy as switching a less than operator into a greater than one.

Limitations

Sometimes AWS S3 would not delete files or delete buckets. My program depends on AWS, so it fails! AWS is not as reliable as I thought it is.

Since I compare files using only their names and modification dates before downloading or uploading, the program might not update some special cases. Consider two directories with the modification date, the same directory structure, and the same file names. The only difference is that those files contain different values.

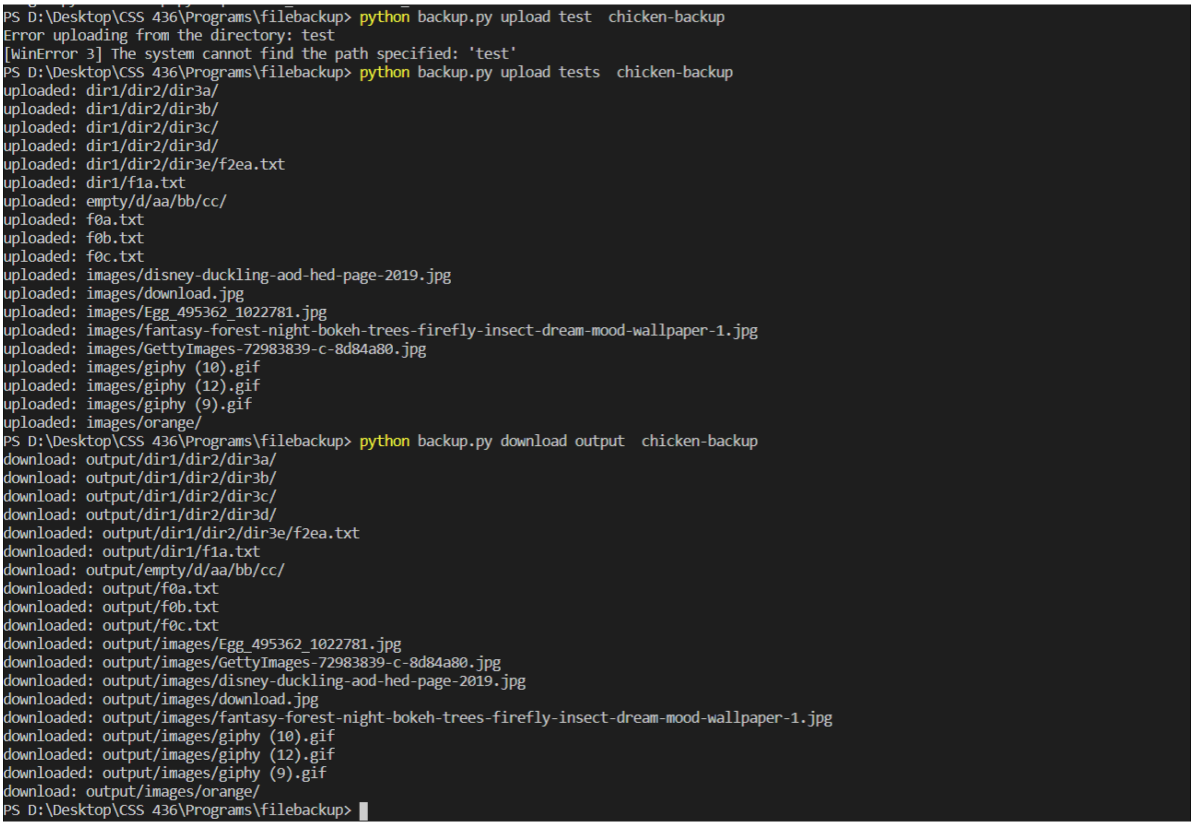
Using my current upload, if you upload directory one to chicken-bucket that will succeed. But try updating directory two to chicken bucket and it will never replace the directory one contents in S3. This is by design because Dr. Dimpsey required that performance improvement. If it were up to me, I would simply clear the bucket every and upload everything every time. That seems like a more expected behaviour. Of course, we can go the Linus Torvalis route and make Git!

Another possible solutions other than using diffs, would be to hash the files and use the hash in comparison. It is not 100% guaranteed to work, but it will, realistically, almost never fail if a suitable hashing algorithm is chosen.

A similar issue happens for download and can be solved using the same techniques.

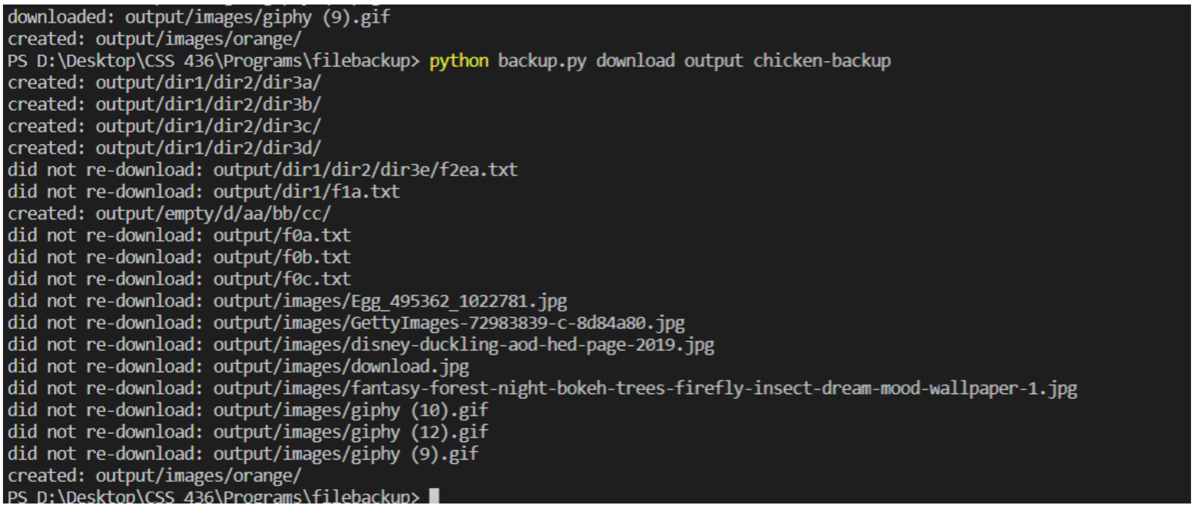
Output

Upload and Download



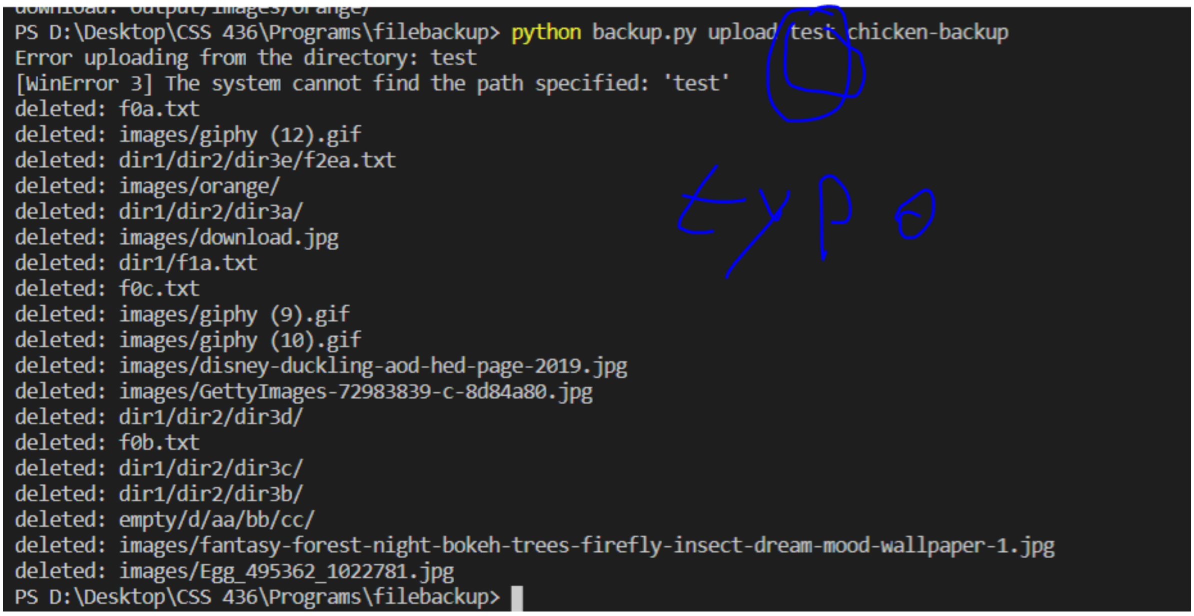
```
PS D:\Desktop\CSS 436\Programs\filebackup> python backup.py upload test  chicken-backup
Error uploading from the directory: test
[WinError 3] The system cannot find the path specified: 'test'
PS D:\Desktop\CSS 436\Programs\filebackup> python backup.py upload tests  chicken-backup
uploaded: dir1/dir2/dir3a/
uploaded: dir1/dir2/dir3b/
uploaded: dir1/dir2/dir3c/
uploaded: dir1/dir2/dir3d/
uploaded: dir1/dir2/dir3e/f2ea.txt
uploaded: dir1/f1a.txt
uploaded: empty/d/aa/bb/cc/
uploaded: f0a.txt
uploaded: f0b.txt
uploaded: f0c.txt
uploaded: images/disney-duckling-aod-hed-page-2019.jpg
uploaded: images/download.jpg
uploaded: images/Egg_495362_1022781.jpg
uploaded: images/fantasy-forest-night-bokeh-trees-firefly-insect-dream-mood-wallpaper-1.jpg
uploaded: images/GettyImages-72983839-c-8d84a80.jpg
uploaded: images/giphy (10).gif
uploaded: images/giphy (12).gif
uploaded: images/giphy (9).gif
uploaded: images/orange/
PS D:\Desktop\CSS 436\Programs\filebackup> python backup.py download output  chicken-backup
download: output/dir1/dir2/dir3a/
download: output/dir1/dir2/dir3b/
download: output/dir1/dir2/dir3c/
download: output/dir1/dir2/dir3d/
downloaded: output/dir1/dir2/dir3e/f2ea.txt
downloaded: output/dir1/f1a.txt
download: output/empty/d/aa/bb/cc/
downloaded: output/f0a.txt
downloaded: output/f0b.txt
downloaded: output/f0c.txt
downloaded: output/images/Egg_495362_1022781.jpg
downloaded: output/images/GettyImages-72983839-c-8d84a80.jpg
downloaded: output/images/disney-duckling-aod-hed-page-2019.jpg
downloaded: output/images/download.jpg
downloaded: output/images/fantasy-forest-night-bokeh-trees-firefly-insect-dream-mood-wallpaper-1.jpg
downloaded: output/images/giphy (10).gif
downloaded: output/images/giphy (12).gif
downloaded: output/images/giphy (9).gif
download: output/images/orange/
PS D:\Desktop\CSS 436\Programs\filebackup>
```

Re-downloading. For objects that end with "/", these are directories. They were simply created if they are not already there. os.mkdirs() handles that logic.



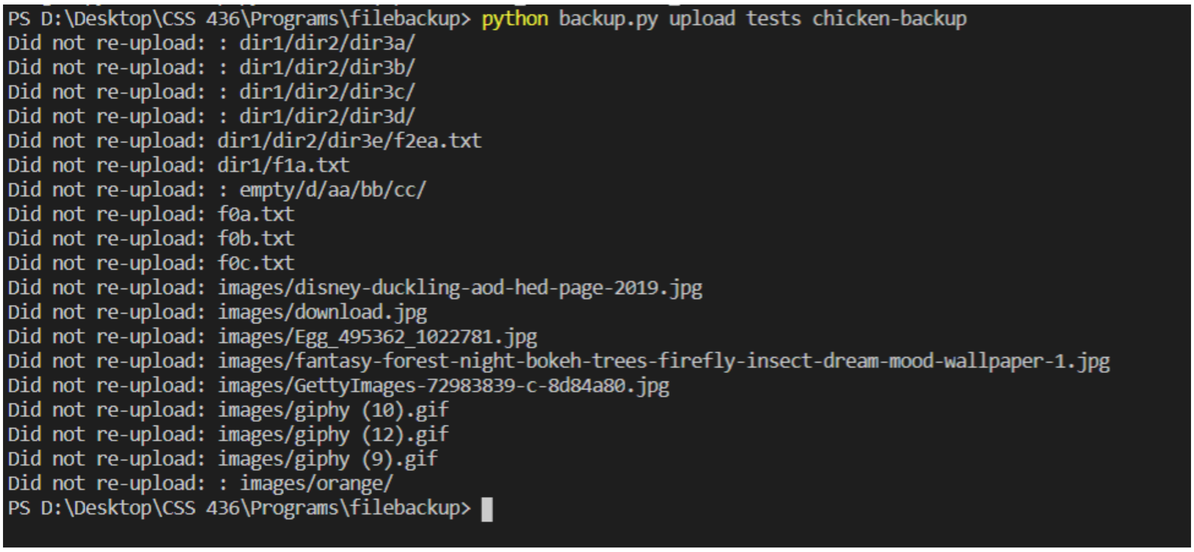
```
downloaded: output/images/giphy (9).gif
created: output/images/orange/
PS D:\Desktop\CSS 436\Programs\filebackup> python backup.py download output chicken-backup
created: output/dir1/dir2/dir3a/
created: output/dir1/dir2/dir3b/
created: output/dir1/dir2/dir3c/
created: output/dir1/dir2/dir3d/
did not re-download: output/dir1/dir2/dir3e/f2ea.txt
did not re-download: output/dir1/f1a.txt
created: output/empty/d/aa/bb/cc/
did not re-download: output/f0a.txt
did not re-download: output/f0b.txt
did not re-download: output/f0c.txt
did not re-download: output/images/Egg_495362_1022781.jpg
did not re-download: output/images/GettyImages-72983839-c-8d84a80.jpg
did not re-download: output/images/disney-duckling-aod-hed-page-2019.jpg
did not re-download: output/images/download.jpg
did not re-download: output/images/fantasy-forest-night-bokeh-trees-firefly-insect-dream-mood-wallpaper-1.jpg
did not re-download: output/images/giphy (10).gif
did not re-download: output/images/giphy (12).gif
did not re-download: output/images/giphy (9).gif
created: output/images/orange/
PS D:\Desktop\CSS 436\Programs\filebackup>
```

Uploading folders that don't exist clears the bucket.



```
download: output/images/orange/
PS D:\Desktop\CSS 436\Programs\filebackup> python backup.py upload test chicken-backup
Error uploading from the directory: test
[WinError 3] The system cannot find the path specified: 'test'
deleted: f0a.txt
deleted: images/giphy (12).gif
deleted: dir1/dir2/dir3e/f2ea.txt
deleted: images/orange/
deleted: dir1/dir2/dir3a/
deleted: images/download.jpg
deleted: dir1/f1a.txt
deleted: f0c.txt
deleted: images/giphy (9).gif
deleted: images/giphy (10).gif
deleted: images/disney-duckling-aod-hed-page-2019.jpg
deleted: images/GettyImages-72983839-c-8d84a80.jpg
deleted: dir1/dir2/dir3d/
deleted: f0b.txt
deleted: dir1/dir2/dir3c/
deleted: dir1/dir2/dir3b/
deleted: empty/d/aa/bb/cc/
deleted: images/fantasy-forest-night-bokeh-trees-firefly-insect-dream-mood-wallpaper-1.jpg
deleted: images/Egg_495362_1022781.jpg
PS D:\Desktop\CSS 436\Programs\filebackup>
```

Reuploading



```
PS D:\Desktop\CSS 436\Programs\filebackup> python backup.py upload tests chicken-backup
Did not re-upload: : dir1/dir2/dir3a/
Did not re-upload: : dir1/dir2/dir3b/
Did not re-upload: : dir1/dir2/dir3c/
Did not re-upload: : dir1/dir2/dir3d/
Did not re-upload: dir1/dir2/dir3e/f2ea.txt
Did not re-upload: dir1/f1a.txt
Did not re-upload: : empty/d/aa/bb/cc/
Did not re-upload: f0a.txt
Did not re-upload: f0b.txt
Did not re-upload: f0c.txt
Did not re-upload: images/disney-duckling-aod-hed-page-2019.jpg
Did not re-upload: images/download.jpg
Did not re-upload: images/Egg_495362_1022781.jpg
Did not re-upload: images/fantasy-forest-night-bokeh-trees-firefly-insect-dream-mood-wallpaper-1.jpg
Did not re-upload: images/GettyImages-72983839-c-8d84a80.jpg
Did not re-upload: images/giphy (10).gif
Did not re-upload: images/giphy (12).gif
Did not re-upload: images/giphy (9).gif
Did not re-upload: : images/orange/
PS D:\Desktop\CSS 436\Programs\filebackup>
```