Assignment 6: Python Scripting

For this assignment, you will use **Python3** to create an alternative to the `rm` command. This program, `rm.py`, will take an arbitrary number of paths as arguments and, for each argument, move them to ∼**/rm_trash** . If ∼**/rm_trash** does not already exist, it should be created.

Hint: The `shutil` module may prove useful.

Duplicate Files

If a file with the same name already exists in ∼**/rm_trash** , you should append **-1** to the end of the filename **before any extension(s) if they exist** so that both files are preserved. If a file with the same name in ∼**/rm_trash** already has **-1**, the new file should be appended with **-2** and automatically incremented for subsequent copies.

For example, if a file named `file.txt` is removed 4 times, ∼**/rm_trash** should contain the following four files: `file.txt`, `file-1.txt`, `file-2.txt`, and `file-3.txt`.

Note: It is possible that a file being removed will already have a dash and a number at the end of the name. If this is true, you should be sure not modify the original name and only append the counter. For example, if a file named `newfile-1.txt` is deleted twice, ∼**/rm_trash** should contain `newfile-1.txt` and `newfile-1-1.txt`. **You may assume that there will be no conflicts in file names (e.g., rm.py file-1.txt; rm.py file.txt)**

Recursion

Your program should recursively delete files if **any** argument is **-r**. If **-r** is not set, directories should be ignored and your program should print
rm.py: cannot remove '*DIR*': Is a directory
to **STDERR** once for each directory encountered where *DIR* is the name of the directory.

When deleting recursively, simply move the directory to ∼**/rm_trash** using the duplicate naming rules as described above and leave the contents alone. You do not need to modify the names of the files **within** the removed directory.

Paths

If an argument does *not* point to a file, your program should print
rm.py: cannot remove '*FILE*': No such file or directory
to **STDERR** once for each invalid argument where *FILE* is the path given.

When moving a file into ∼**/rm_trash** you should always place it at the root of the directory. For example, if `/path/to/a/file.txt` is passed as an argument, `file.txt` should be moved to ∼**/rm_trash/file.txt** and *not* ∼**/rm_trash/path/to/a/file.txt**.

Program Execution

Your program should be invoked as a single Python file (see below) with an **arbitrary** number of arguments representing paths to files (including directories). The **-r** option may be passed as *any* argument. Some examples are listed below.

```
$ rm.py /path/to/some/file ./somefile someotherfile
$ rm.py /path/to/some/file ./somefile someotherfile -r
$ rm.py -r /path/to/some/file ./somefile someotherfile
$ rm.py *.java
```

Extra Credit (10 points)

For extra credit, create a separate program called **restore.py** which, given a path to where a file *used to be* before it was removed with **rm.py**, will restore the *most recent version*.

Hint: Consider keeping track of the original and new locations of removed files to handle situations where removed files from different paths have the same name. Note that if you use this strategy, the information will need to persist between executions of **rm.py** and **restore.py**.

Attempts at extra credit will **not be considered** if **rm.py** does not work correctly.

Assignment Data

No sample data is included with this assignment as it should work for all files. As always, you should test your program with your own data as well to be sure that it works correctly.

Program Files

Your submission will consist of one or two files depending on if you attempt the extra credit:

- **rm.py** - implementation of rm

- **restore.py** - extra credit

Submission

Turn your assignment in via Blackboard. Your zip file, named **LastnameFirstname.zip** should contain only your **rm.py** and optionally **restore.py**.

If you attempt the extra credit, name your file **LastnameFirstname_EC.zip**. Without the **_EC**, your submission will be graded as normal.