Elif Nur Akalın
2018400312
CMPE 230
21 May 2020

# Identical Finder in Python

In this project, we are asked to write a command line python program that finds the identical files or directories in any folder. They can be identical in contents or names. They are printed with their full path names and with or without their sizes.

First, I implemented the arguments handling with the Argparse package. It has a mutually exclusive group, consisting of "-f" and "-d", the arguments for searching for files or directories, respectively. It has "-c", "-n" arguments, the former is the argument for searching for identical contents and the latter for identical names. They can be used at the same time, resulting in finding identical-named items with identical contents. There is "-s argument, which is for printing the output with their sizes.

The program compares items according to their hash values, which are calculated using the SHA256 encryption.

In the *searchfiles* method, which searches for files recursively. For all files, if only -n is given as argument, it adds to the dictionary only the name as key and path and size as values. If no argument is given, it pretends like -c argument is given. It hashes every file, and adds them to the dictionary. If the hash exists in the dictionary, it appends the values to the existing values of that hash. If both -c and -n are given arguments, the filename is hashed and concatenated with the hash, and hashed again.

In the *searchdirsname* method, which is called only when directories are searched for identical names. The loop inside the method calls recursively and checks the name of every directory, hashes them, adds them to the dictionary of hashes.

In the *searchdirs* method, which works similarly to searchfiles method. The difference is in the recursion. First all the files inside that folder (and not the files inside the subfolders) are listed and hashed. They are added to a list called dirhashes. It is the dictionary for holding the hash values of the files and folders immediately inside that folder. After that, for every directory inside the current one, searchdirs is called again and recursion is created. This way, until the method is in a leaf directory, the method is called. Then, all the hashes in the dirhashes are concatenated together and this string is hashed again. This hash is then added to hashes dictionary.

For output, my program prints the full paths of the identical items. If "-s" argument is given, their sizes are also printed, and the items are sorted according to their sizes.

In conclusion, my program executes appropriately. There are no errors and works perfectly in all cases.