CMPE230 SYSTEMS PROGRAMMING

HOMEWORK 2

1.PROBLEM DESCRIPTION

We are asked to implement a python program named identic which will traverse the directories and look for files or directories that are duplicates of each other. The full pathnames of duplicates will be printed as output. Search criteria for the duplicate files/directories must be determined according to the arguments received from the terminal and program must be run from the terminal using the format "python identic.py [-f | -d] [-c] [-n] [-s] [<dir1> <dir2> ..]".

2.IMPLEMENTATION DEATILS

Program is implemented as single class named “identic” with extra 4 methods. Uses sha256 for hashing to determine identical files/directories. A logic like hash tree logic is selected to hash directories. Contains global variables such as:

Hash_dict: a dictionary that stores hash values as keys and set of paths as value.

Args: returned object of argument parser.

Contents and names: lists that contain current directory’s children’s hash values.

Directories: list that holds value of given directory list after discarding subdirectories and getting absolute path of every path.

Child_directory: dictionary that hold hash values of calculated children. In this way, recalculation is avoided. It holds path string as key and list of hash values (name, content, size) as value for each calculated directory. Size of the list depends flags.

First, determines the operation criteria of the program by reading the arguments from the terminal. Argument reading is done using argparse library. “-f” and “-d” added as mutually exclusive since using them at the same time is not allowed. -cn option is same as -c and -n option so -cn option is not added as extra argument. When adding arguments, comments are added to the help option to make program more user friendly. It is assumed that size flags can true only when content flag is true. If no option selected between -c and -n, since -c option is default one, sets content flag true. If no directory is given current directory is used. Subdirectories are removed from provided list and all paths are converted to absolute paths.

If -d is given as argument, program searches duplicate directories in the given directory list. All directories and files are traversed by using os.walk. After calculating the hash of each directory, the values are put into the dictionary named child_directory. In this way, while calculating hash of the parent directory, the values of the previously calculated directory hashes are not recalculated. Since each directory will have one direct parent file, the file whose value is used is deleted from the dictionary to reduce the space complexity.

If directory flag is false, searches for identical files in directory list. In this way -f option becomes default value of file/directory group. All files in each given directory is traversed by using os.walk. If necessary, hash file's content by calling hash_file method, calculates file's size and hashes file's name. Updates dictionaries by calling update_dict method.

The values included in the dictionary and list are determined by the arguments entered. For example, if the content flag is true, hash value of content is calculated and added to the lists and dictionaries. In this way, unnecessary calculations are prevented. If -cn option is given, hash of content and hash of name concatenated and used as key in hash dictionary.

After traversing all directories, sorts each duplicate set according to lexicographic order. If -s option is selected, prints duplicate sets according to size in decreasing order, if the size of the two duplicate sets are the same, they are sorted alphabetically. Else prints duplicate sets according to lexicographic order. First element is used to determine order between duplicate sets. Prints blank line between each duplicate sets.

1.get_abs (d_path)

Returns absolute path of given relative or absolute path, raises error if given path is not valid. Paths that starts with ".", "..", "~" also considered as valid.

2.hash_file (file_path)

Takes absolute path of file as parameter. Opens the file with the byte option to read files with different extensions. In case the size of the file is large, it is preferred to read block by block instead of reading the whole file at the same time. Returns hash value of file's content.

3.parent (child_hash, parent_hash)

Takes list of children's hash values and initial hash of parent as parameter. Sorts the hash list before concatenation to avoid problems that may rise from the order. Then concatenates elements of the list and initial value of parent's hash. Returns hash of result as parent's final hash value.

4.update_dict (key, value)

This method is used to update hash dictionary. Takes key and value as parameter. The content of the dictionary was already mentioned.

3.CONCLUSION

The program contains all features which are specified in the project description. To determine duplicates, it hashes files/directories using sha256 and prints duplicates sets in the desired order. The hash values of a folder's and an empty file's contents are considered the same. It is aimed to implement a program that can run on both windows and linux operating systems in efficient way.

İrem Zeynep ALAGÖZ 2018400063