#!/bin/bash

# check if the script was run with an argument

if [ -z "$1" ]; then

echo "Please provide a search keyword as an argument."

exit 1

fi

# set the search keyword from the argument

keyword="$1"

# search for files in the directory matching the keyword and store their names in an array

files=($(find . -name "\*$keyword\*" -type f))

# set the name of the output file

output="diff.txt"

# loop through the array of files and use the diff command to compare them

for (( i=0; i<${#files[@]}-1; i++ )); do

for (( j=i+1; j<${#files[@]}; j++ )); do

diff "${files[i]}" "${files[j]}" | grep "^<" | sed 's/^< //' >> "$output"

diff "${files[i]}" "${files[j]}" | grep "^>" | sed 's/^> //' >> "$output"

done

done

echo "Differences saved to $output"

In the script, files[@] refers to the array of file names that were found by the find command. The @ symbol means that the array should be expanded to include all its elements, so ${#files[@]} returns the number of files in the array.

diff "${files[i]}" "${files[j]}" | grep "^<" | sed 's/^< //' >> "$output"

diff "${files[i]}" "${files[j]}" | grep "^>" | sed 's/^> //' >> "$output"

use the diff command to compare the contents of two files, ${files[i]} and ${files[j]}, and then use grep to filter out only the lines that start with < and > respectively. Finally, the sed command is used to remove the < and > characters at the beginning of each line. The resulting output is then appended to the file specified by the output variable. This effectively saves the differences between all pairs of files in the files array to the file diff.txt.

The grep command with the ^< and ^> patterns is used to filter the output of diff to only include the lines that differ. The ^< pattern matches lines that start with <, and the ^> pattern matches lines that start with >. The sed command is then used to remove the < and > characters from the output, leaving only the differing lines.