OldNotInNew and NewNotInOld

Problem Statement

I back up files monthly. After each back up, I compare the *new* backup against the *old* to determine:

- which *new* files have been added since the old backup, and
- which *old* files found in the old backup are missing from the new backup.

Desired Output

Create a program which writes these two text files based upon the input described in Provided Input.

`OldNotInNew.txt`	each line lists a file found only in the old backup (not in new backup)
`NewNotInOld.txt`	each line lists a file found only in the new backup (not in the old backup)

What to Submit

Source code for your solution, along with instructions on how to build and run. (Feel free to use any scripting language or programming language that is available on Windows.)

Provided Input

The program shall take as input two text files (you may assume ASCII with Windows line ending):

`Old.sha1.txt`	lists all files, along with their SHA-1, found in the old backup
`New.sha1.txt`	lists all files, along with their SHA-1, found in the new backup

Each line in `Old.sha1.txt` and `New.sha1.txt` contains the following:

- the hexadecimal representation of the file's SHA-1 hash (40 characters),
- a single space character, and
- the filename of the file that generated the SHA-1.

Here is an example of a line that might be found in `Old.sha1.txt` or `New.sha1.txt` (or both):

```
6905d2e6c152e75b9a2870f99e9d953531ba62a7 C:\cache\Hazlitt Economics in One Lesson.pdf
```

Additional Requirements

Files comparison must be made based only on their SHA-1:

- Two files with the same SHA-1 are considered to be the same file
  - Even if the filenames differ
- Two files with identical filenames, but different SHA-1 hashes, are considered to be different

Your program must be able to process `Old.sha1.txt` and `New.sha1.txt` files each which may contain up to 500,000 lines (approximately 80 MB). (You may support processing more lines if you wish.)

Your program must support up to 260 characters long filenames. (You may support longer if you wish.)

Filenames may contain spaces, may contain forward slash or backslash, and may or may not be quoted.

No assumptions may be made about the ordering of lines within an input file.

Example

The old backup contained two files: `file 1.txt` and `file 2.txt`.

The new backup contains three files: `file 8.txt`, `file 2.txt` and `file 3.txt`.

Example: Input

The backed-up files and their SHA-1 hashes are listed in these two text files:

`Old.sha1.txt` contains:

```
1111111111111111111111111111111111111111 file 1.txt
2222222222222222222222222222222222222222 file 2.txt
```

`New.sha1.txt` contains:

```
1111111111111111111111111111111111111111 file 8.txt
7777777777777777777777777777777777777777 file 2.txt
3333333333333333333333333333333333333333 file 3.txt
```

Example: Expected Output

Given the above input, your program should write the following two text files:

`OldNotInNew.txt` contains:

```
file 2.txt
```

`NewNotInOld.txt` contains:

```
file 2.txt
file 3.txt
```

Note that the `file 2.txt` listed in `Old.sha1.txt` and `New.sha1.txt` refers to two different files:

- The `file 2.txt` listed in `OldNotInNew.txt` refers to `Old.sha1.txt`'s `file 2.txt`, which has a SHA-1 of `2222222222222222222222222222222222222222`. No files in `New.sha1.txt` contains this SHA-1, which is why `file 2.txt` is listed in `OldNotInNew.txt`.
- The `file 2.txt` listed in `NewNotInOld.txt` refers to `New.sha1.txt`'s `file 2.txt`, which has a SHA-1 of `7777777777777777777777777777777777777777`. No files in `Old.sha1.txt` contains this SHA-1, which is why `file 2.txt` is listed in `NewNotInOld.txt`.

Note that neither `file 1.txt` nor `file 8.txt` are listed, as the same SHA-1 exists in both `Old.sha1.txt` and `NewNotInOld.txt`.

Notes

Information about SHA-1: https://en.wikipedia.org/wiki/SHA-1

To generate your own `New.sha1.txt` and `Old.sha1.txt` test data, you may wish to use `fciv.exe` (download for Windows), `sha1sum` (Cygwin; may need to fix line ending so they're Windows format), or `Get-FileHash` (PowerShell 4.0; however, you will need to format the output to match above).