

Forum:[Hack Forums](#)

Thread: [Rate my "md5 hash every line in text into a new file" script.](#)

Hammylein - March 18, 2022, 1:48 am

Good Morning to y'all from a python beginner,

I was just trying to turn a text file full of words into md5 to then test said md5'd file against my hashcat rule ideas.

This is python script i came up with to turn my entire plaintext file into a md5 text file.

```
Code import hashlib
inFileName = input('Please type input filename:')
outFileName = input('Please type output filename:')
outFile = open(outFileName, 'w')
```

```
with open(inFileName) as f:
    lines= [line.rstrip() for line in f]
for line in lines:
    md5 = hashlib.md5(line.encode())
    md5hash = md5.hexdigest()
    print(line + " : "+md5hash)
    outFile.write(md5hash+'\n')
outFile.close()
```

What could i improve visually if at all. Functionally it works as intended.

Cheers

Faith Connors - March 20, 2022, 10:40 am

imo:
--take filename in from args (i.e. filename.txt, and make the output file filename_md5.txt
--is the print() on line 11 actually needed? do note that ive had stdout be a bottleneck for my code (i.e. removing print statements has made my code run much faster in the past, specifically in old compsci assignments.)

looks clean otherwise. short and simple. there is one inconsistent naming convention (f vs outfile, despite both representing a file handle)
i dont like how you use operators lol, for the print() statements, the first '+' is surrounded with spaces to pad it and make it look nice, but the second '+' is just cramped in for no reason (and that style is repeated in the next line)

if you get rid of the print() statement, you may as well also get rid of assigning the digest to 'md5hash' since you're only using it in the write() call anyway.

3phase - March 23, 2022, 7:11 am

(Mar 18, 2022 01:48 AM)Hammylein Wrote: [CITE](#) Good Morning to y'all from a python beginner,

I was just trying to turn a text file full of words into md5 to then test said md5'd file against my hashcat rule ideas.

This is python script i came up with to turn my entire plaintext file into a md5 text file.

```
Code import hashlib
inFileName = input('Please type input filename:')
outFileName = input('Please type output filename:')
outFile = open(outFileName, 'w')
```

```
with open(inFileName) as f:
    lines= [line.rstrip() for line in f]
for line in lines:
    md5 = hashlib.md5(line.encode())
    md5hash = md5.hexdigest()
    print(line + " : "+md5hash)
    outFile.write(md5hash+'\n')
outFile.close()
```

What could i improve visually if at all. Functionally it works as intended.

Cheers

Well, your code works as intended in a perfect conditions environment.

if you get a wrong input file name (or path), your code crash, but it only crashes after it has already made the output file

which makes no sense, cause the output file will be empty.

Thats my version of your code -

```
Code import hashlib
input_path = input('Input file path:\n')
output_path = input_path.replace('.', '_md5.')
```

```
try:
    with open(input_path, 'r') as f:
        lines = [line.rstrip() for line in f]
        with open(output_path, 'w') as output:
            output.write('\n'.join([hashlib.md5(line.encode()).hexdigest() for line in lines]))
    print('Done')
```

```
except FileNotFoundError:
```

```
    print(f'No such file at path: {input_path}')
```

- The input file path, can be relative to the main.py file or absolute path, both works (but its path in both cases)

- output_path is derived from input_file name, less hassle for the user

- Opening output file only after the program successfully read all lines from the input file (also catching the error if the file does not exist)

- Writing into file written in one line with joined "list comprehension", you decide if you like it or not, it's up to you if you're familiar with the syntax

cremetarte - March 24, 2022, 5:18 pm

Posting here to follow what people say. thanks for sharing the code.

foxegado - March 30, 2022, 12:37 pm

You could always do something like this so that it can do more than just md5...

```
Code from __future__ import annotations
```

```
from hashlib import md5, sha1, sha256
from pathlib import Path
from typing import Callable
```

```
def getFilePath() -> Path:
    fileInput = input('Enter file name: ')
    if (inputPath := Path(fileInput)).is_file():
        return inputPath.absolute()
    print(f'Could not find path {fileInput}. Please try again.')
    return getFilePath()
```

```
def hash(inputStr: str, hashFunction: Callable):
    return hashFunction(inputStr.encode()).hexdigest()
```

```
def genHashFile(inputPath: Path | str, outputPath: Path | str, hashFunction: Callable):
    with open(outputPath, 'w') as out:
        out.writelines([hash(x.rstrip(), hashFunction) + '\n' for x in open(inputPath, 'r').readlines()])
```

```
def main():
    inFile = getFilePath()
    hashTypes = {'md5': md5, 'sha1': sha1, 'sha256': sha256}
    for hashType, hashFunction in hashTypes.items():
        outFile = inFile.with_name(f'{inFile.stem}_{hashType}{inFile.suffix}')
        genHashFile(inFile, outFile, hashFunction)
```

```
if __name__ == '__main__':
    main()
```