

Week 8 - Encrypt and Decrypt Files with Python

Encrypting a file using the cryptography library

Encryption is the process of encoding information so that only authorized parties can access it.

This example uses symmetric encryption. The same key used to encrypt data can be used to decrypt it.

Individual files or file systems are protected by file encryption, which encrypts them with a unique key and makes them available only to the owner of the key.

The goal is to prevent malicious or unauthorized persons from accessing files on the hard drive.

The operating system or file system may provide support for file encryption. Confidential files can only be accessed using the decryption key.

If a user needs to securely transfer individual files over the Internet or store them on a portable medium such as a USB drive, file encryption comes in handy.

Cryptography is a term that refers to the process of encrypting and decrypting data.

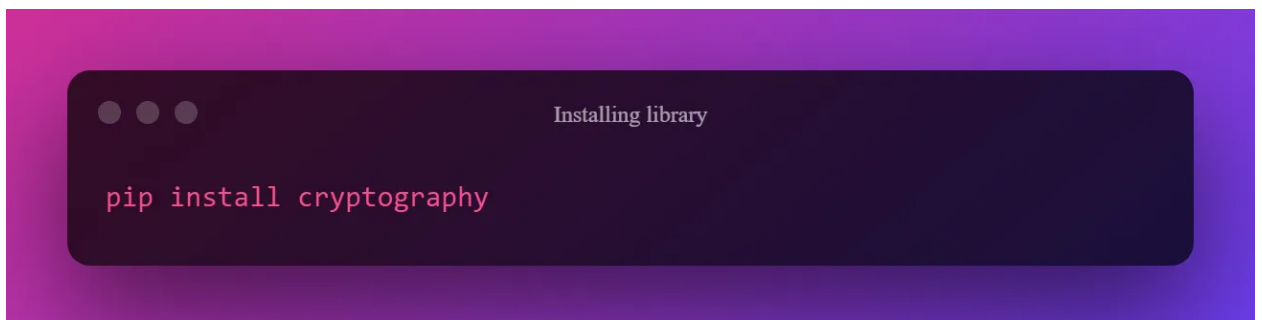
Let's see how we can use Python to encrypt and decode some of our data. We will use symmetric encryption, which means that we will encrypt and decrypt data with the same key.

Steps to encrypt and decrypt a file:

1. Library installation
2. dataset
3. Create a key
4. Key download
5. File encryption
6. File decryption

1. Installing the Library

Please open "Command Prompt" (on Windows) and enter the following code to install them:

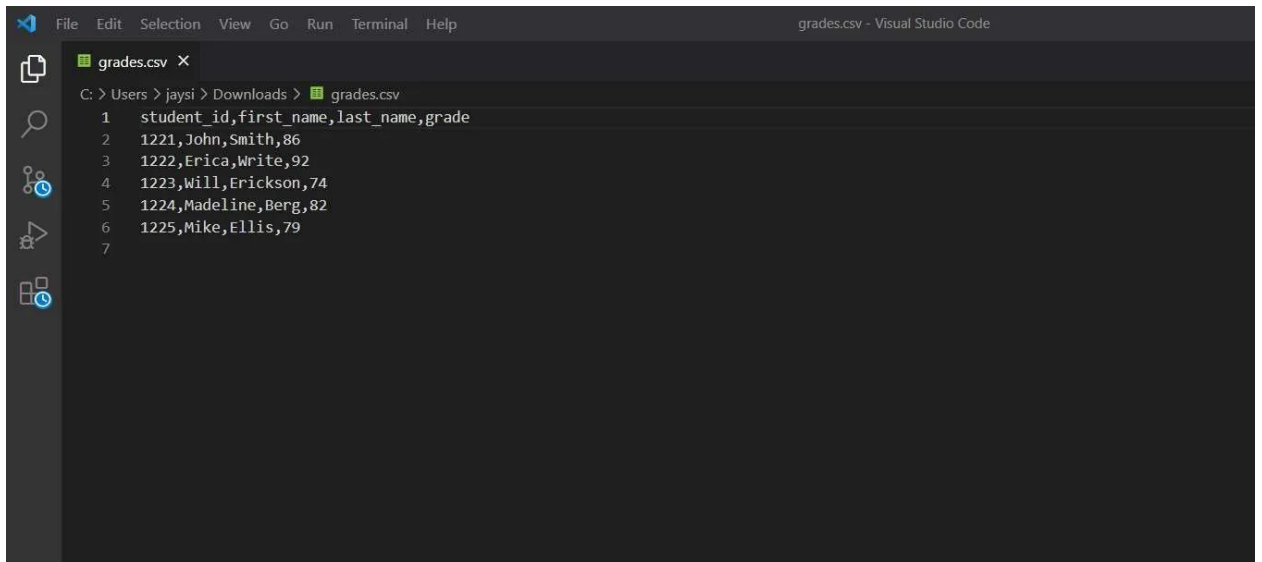


```
Installing library

pip install cryptography
```

2. Data set

First we need an example file. Here [sample](#) CSV file containing student grade information.

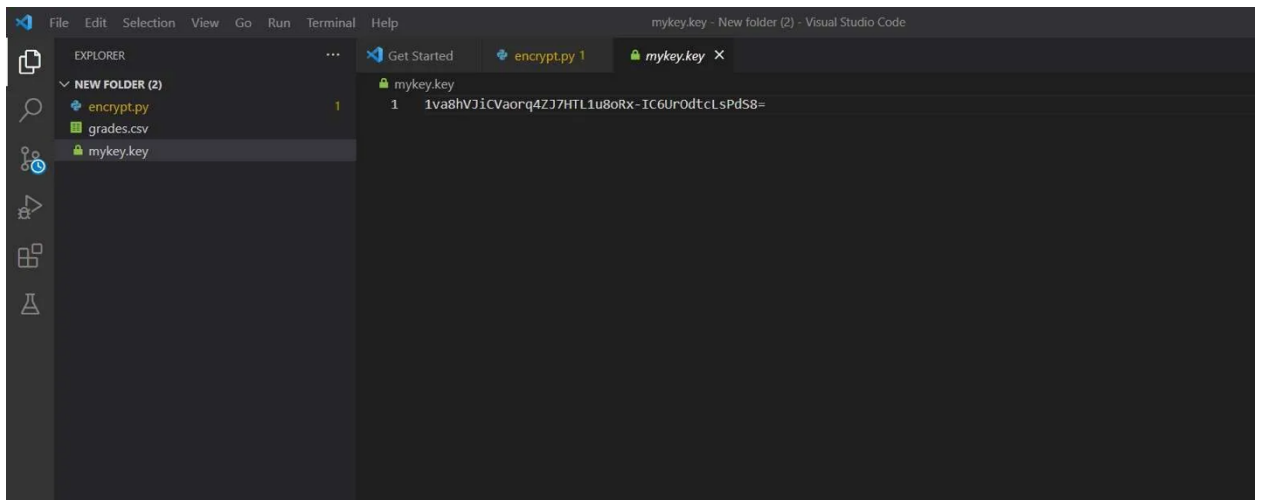


3. Creating a Key

In our example, we will use the symmetric equation. Fernet is a type of authenticated encryption that requires a "key" to read and/or modify a file. Now we will create a key and place it in the same directory as our data file:

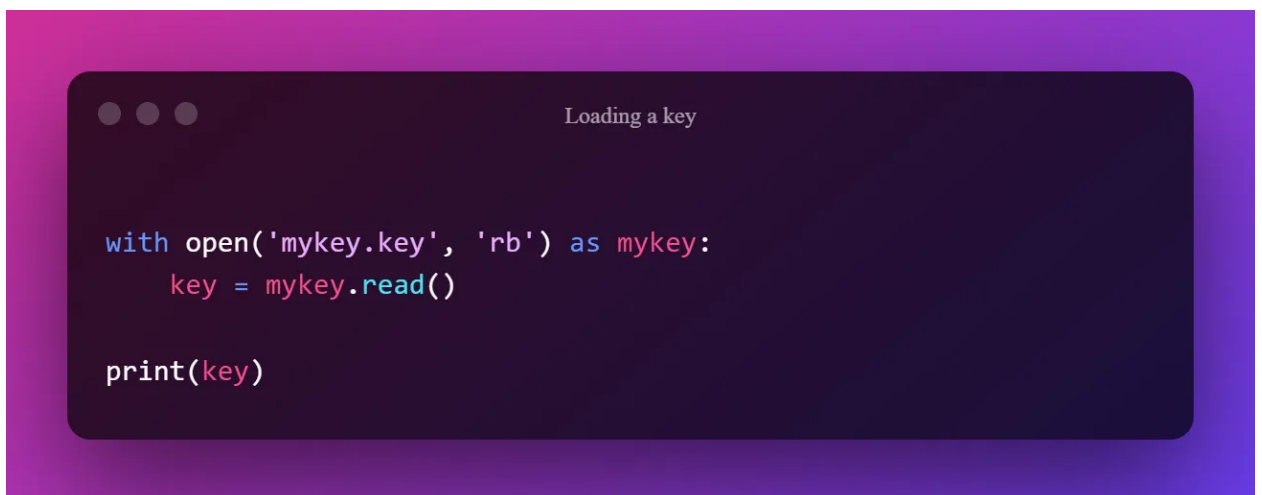


If you go into the directory where your Python code is, you should find the mykey.key file. There should be only one line in the file, which is a string of characters in some sequence. You can look at my key below, but yours will be different.



4. Download Key

We will need to upload the encryption key to our environment once we have generated it in order to encrypt/decrypt the files. The next step is quite simple and only requires opening the mykey.key file and saving it to local memory:



The encryption key is now stored locally as a key variable.

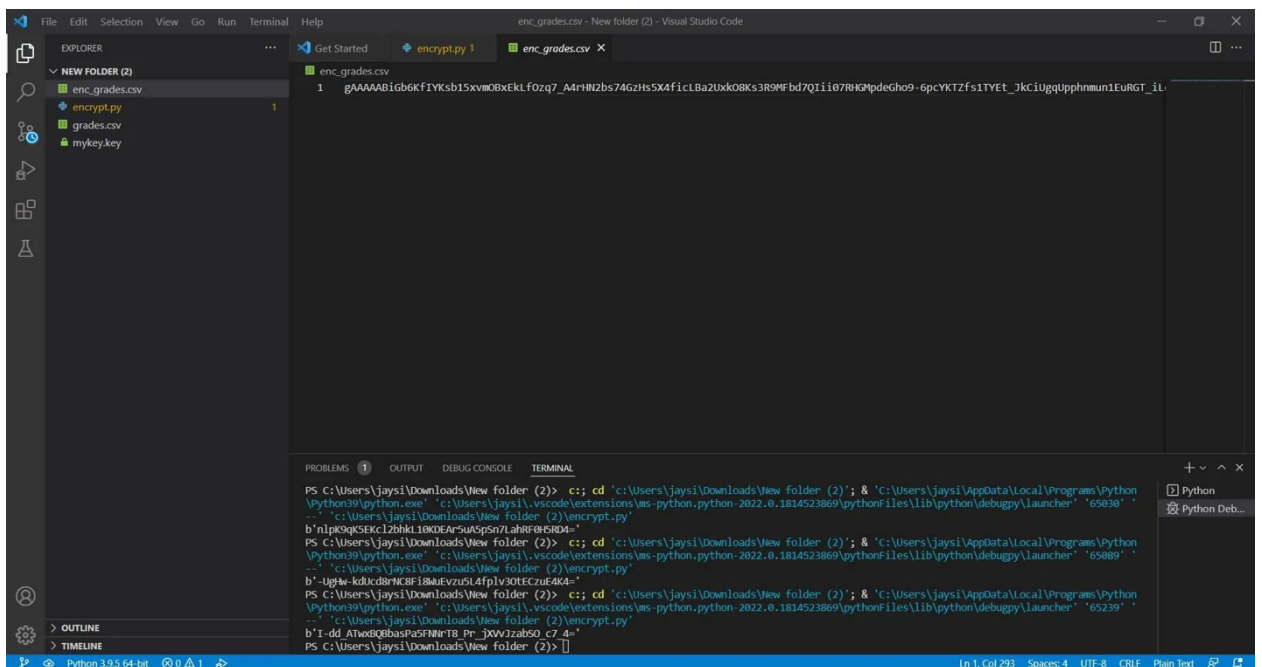
5. File Encryption

We will create a function to use the encryption key and return the encrypted file now that we have the file to encrypt and the encryption key. We save the Fernet object as a local variable `f` when we create it.

After that, we imported our original data (grades.csv) into the original. The data is then encrypted using the Fernet object and stored encrypted.



Finally, we save it as "enc_grades.csv" in the new.csv file. The encrypted file can be viewed here:



6. Decrypting the File

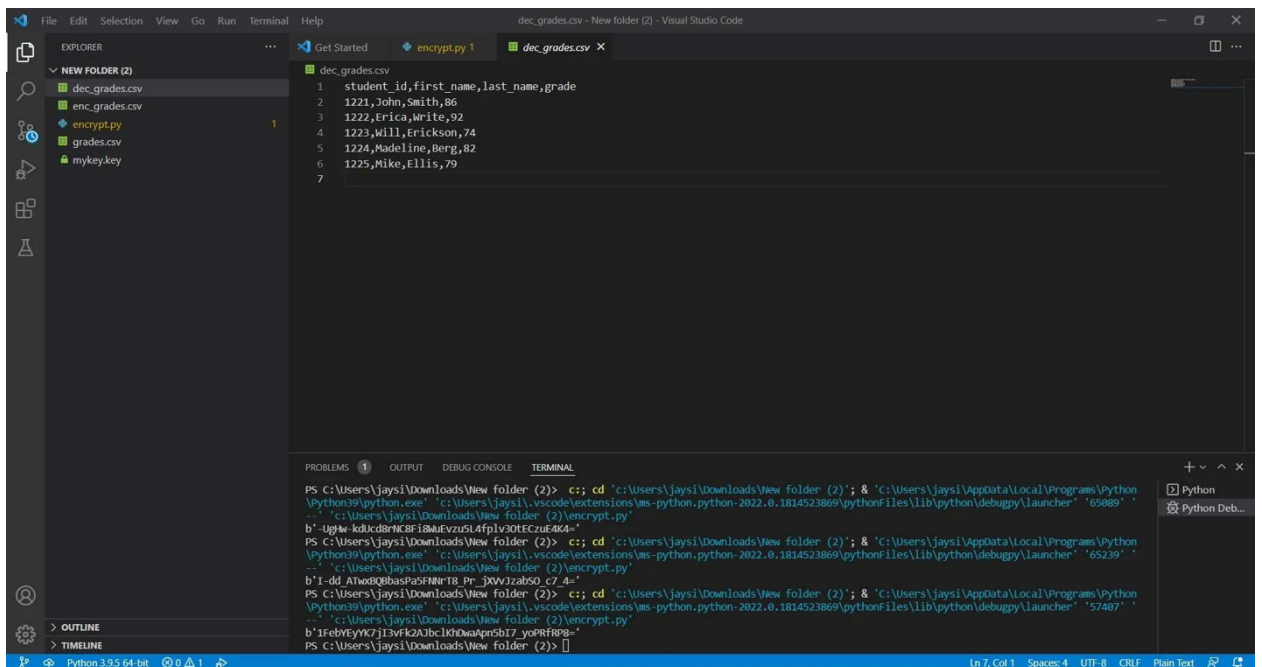
You will want to access the file after you encrypt it and, for example, successfully move it to another location. Now this information is encrypted.

The next step is to recover the original material by transcribing it. The procedure we will use now is the reverse of the encryption procedure we used in the previous section.

We will follow the same steps as before, but this time we will move from an encrypted to a decrypted file:



Finally, we save it as "dec_grades.csv" in the new.csv file. The encrypted file can be seen below:



Total in learned how to encrypt and decode a file and the data it contains using symmetric file encryption type in this post using Python [programming language](#) and a cryptographic package.