In this phase, you'll develop modules that will encrypt/decrypt a file.

You will design these modules:

* Myencrypt(message, key)
* Mydecrypt()
* MyfileEncrypt(filepath)
* MyfileDecrypt(filepath)

(C, IV)= **Myencrypt**(message, key):

1. Generate a 16 Bytes IV.
2. Encrypt the message using the key and IV in CBC mode (AES).
3. Return an error if the len(key) < 32 .
   1. i.e., the key has to be 32 bytes= 256 bits.

(C, IV, key, ext)= **MyfileEncrypt** (filepath):

1. Generate a 32Byte key.
2. Open and read the file as a string.
3. Call the above method to encrypt file using the key generated.
4. Return:
   1. cipher C
   2. IV
   3. key
   4. extension of the file (as a string).

Write the **inverse** of the above methods.

Must be able to:

1. Encrypt a JPEG file.
2. Decrypt the JPEG file.
   1. Make sure that it is still viewable.

Next part of assignment:

* Write this method:
  + RSACipher, C, IV, ext)= **MyRSAEncrypt**(filepath, RSA\_Publickey\_filepath
* Call MyfileEncrypt (filepath)
  + Returns (C, IV, key, ext)
* Initialize an RSA public key encryption object and load pem publickey from the RSA\_publickey\_filepath.
* Encrypt key variable ("key") using RSA publickey in **OAEP** padding mode.
  + The result will be RSACipher.
* Return (RSACipher, C, IV, ext).
* Do the inverse (MyRSADecrypt (RSACipher, C, IV, ext, RSA\_Privatekey\_filepath))
  + Does the exactly inverse of the above.
  + Generate decrypted file using previous decryption methods.

Use Github to commit and push all of your code so the instructor can see your source.