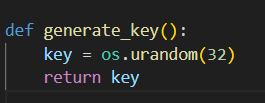
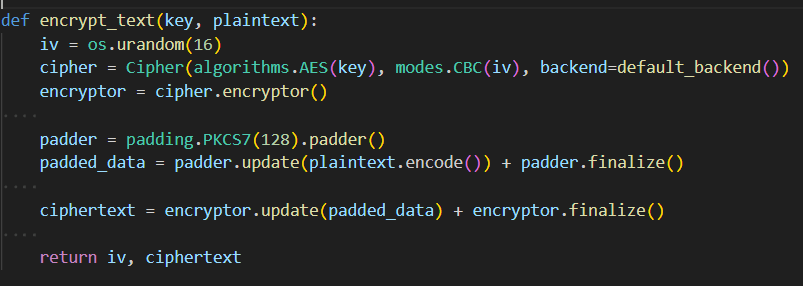
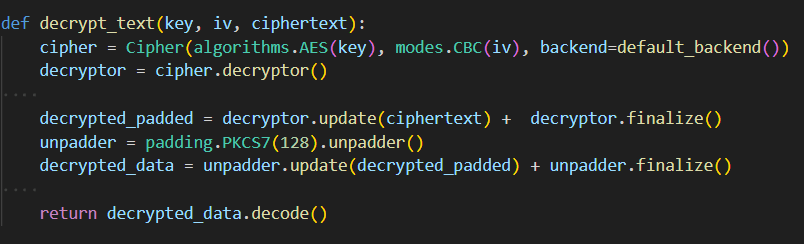
GIT REPO - [git@github.com:Cabba4/aliceAndBob.git](mailto:git@github.com:Cabba4/aliceAndBob.git) (private but can add access if requested)

Exercise 1:

*   
  Key is generated using urandom() function with 32 bytes = 256 bits for AES-256
*   
  User input text is in plaintext and key is generated from generate\_key function.
*   
  Similarly decrypt function uses the same padding, mode and algorithm to decrypt.
* A screen shot of a computer program

  Description automatically generated
* There is a significant time difference between encryption and decryption functions which was persistent even after running the script multiple times. In theory both operations should take similar times, the variations can be because of improper time measurements from python time module and inner workings of the processor.
* EMAILED - Ciphertext : w/JXRjpurylpN5BDd09x2PwSrU/o8ilf+m4hX7XIxubTL7gOxdRiZLaoWybTQg4noHr+9m5jE0/N5tj5CHI2BA==  
  EMAILED key : YoLVv0PJ7RDLKMrkvIqS0aVBdrFjaFmHuifPV+eFjZ0=

Exercise 2

* A computer screen shot of text

  Description automatically generated  
  Function to create rsa keys for given username – omitted check for existing possible keys
* A black screen with white text

  Description automatically generated  
  A computer screen with text

  Description automatically generated  
  Encryption function using public\_key and plaintext
* A computer screen with white text and blue text

  Description automatically generated  
  Decryption function using private\_key
* A black screen with white text

  Description automatically generated  
  Encryption is a bit faster than decryption in rsa because encryption involves fewer computations. Decryption involves modular exponentiation with private key and can be slower due to larger key size. Private key > public key.
* A computer screen with text on it

  Description automatically generated  
  Difference between encryption and decryption is minimal. However now we need to generate keys for both alice and bob and then create extra functions to derive shared sending and receiving keys.

Exercise 3

* A screen shot of a computer program

  Description automatically generated
* Function to compute sha256 hash of provided file
* Also included function to validate integrity of received file.
* Received file – sample\_text.text with contents   
  “This is a simple paragraph of text for hash verification purposes.”
* Received hash - c124ed65c5b9c1bc99f7cdc09553f286e59eea1e8a456976d028761f14d01669
* Integrity check and hashing my text file -   
  sample.txt – “This is a simple paragraph of text for hash verification purposes.”  
  hash - 306a6cd313a2148e41e517a94e3fe6e899ccbeefa64edc001b5da27f8f7238a6
* A computer screen shot of a program

  Description automatically generated
* First one with sample\_text.txt and comparing with hash provided by course mate. Then running with sample.txt which is generated by the generate\_text() function.
* Making small change in sample.txt  
  A computer screen with white and blue text

  Description automatically generated  
  Only adding two “!!” entirely changed the sha256 digest.