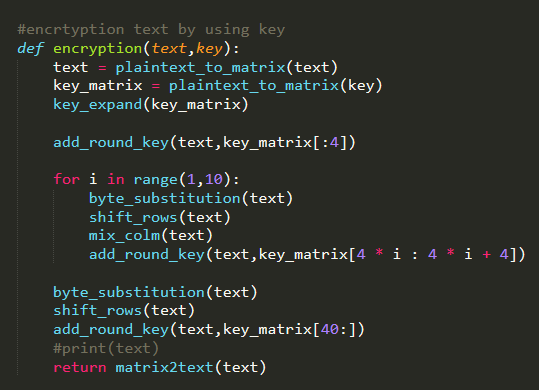
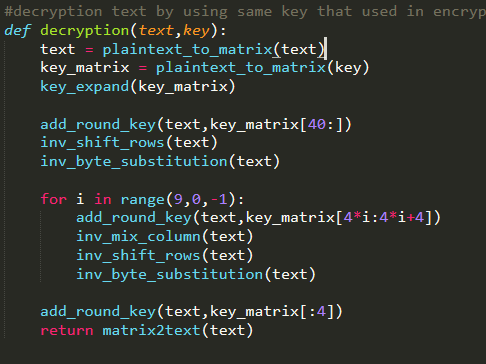
Part1) Benim algoritmamda AES 128 bit plaintext ve 128 bit key değeri alıyor.

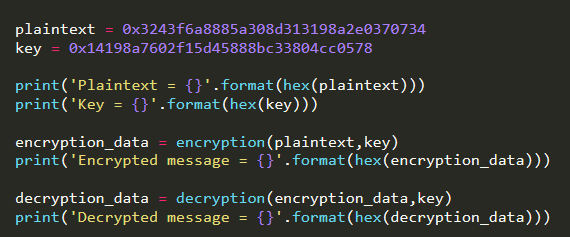


AES encryption algoritmasında önce plaintexti ve anahtarı bir 4x4 matrise dönüştürüyorum. Sonra anahtarı key\_expand ile genişletiyorum ve anahtarı 41 elemanlı bir matrise dönüştürüyorum. Daha sonra klasik AES’nin roundları başlamış oluyor. İlk olarak add\_round\_key algoritması uygulanıyor. Daha sonra 9 round boyunca for döngüsü içerisinde sırasıyla byte\_substitution, shift\_rows, mix\_column ve add\_round\_key algoritmaları gerçekleşiyor. Son olarak AES algoritmasında olduğu gibi byte\_substitution,shift\_rows ve add\_round\_key algoritmalarını bir kez daha uyguluyorum. En son algoritma başında oluşturduğum plaintext matrisini hexadecimal formata çevirip return ediyorum.

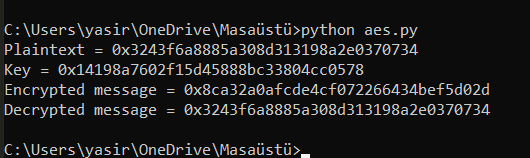


Decryption algoritmasıda encryptiona çok benzer. Tekrar algoritma başında matrislerimi oluşturup anahtarı genişletiyorum. Sonra add\_round\_key, Shift\_rowun tersi ve byte\_substitution işleminin tersi algoritmaları uyguluyorum. Daha sonra roundlarım for döngüsüyle başlıyor ama bu sefer encryption aksine döngüm ters dönüyor. Döngüde yapılan işler encryptiondakilerle aynı sadece add\_round\_key fonksiyonu dışındaki diğer fonksiyonların ters algoritmaları çalışıyor. En son AES decryption yapısında olduğu gibi add\_round\_key işlemini yapıp matrisimi hexadecimal formata çevirip return ediyorum.

Örnek bir testi inceliyecek olursak :



Plaintextimi ve anahtarımı 128 bit olucak şekilde hex formatında AES nin encryption ve decryption fonksiyonlarını çağırığ test ediyorum.



Programı çalıştırdığımda ekran görüntüsünde ki çıktıyı alıyorum. Önce plaintext ve anahtarı ekrana yazdırıyorum daha sonra Encrypt edilmiş halini yazdırıyorum. Kodumda bu encrypt edilmiş halini decrypt fonksiyonuna gönderiyorum ve çıkış olarak aldığım değer görüldüğü üzere ilk plaintext değerimle aynı oluyor.

Kodumu Python 3.8.5 versiyonun da yazdım.