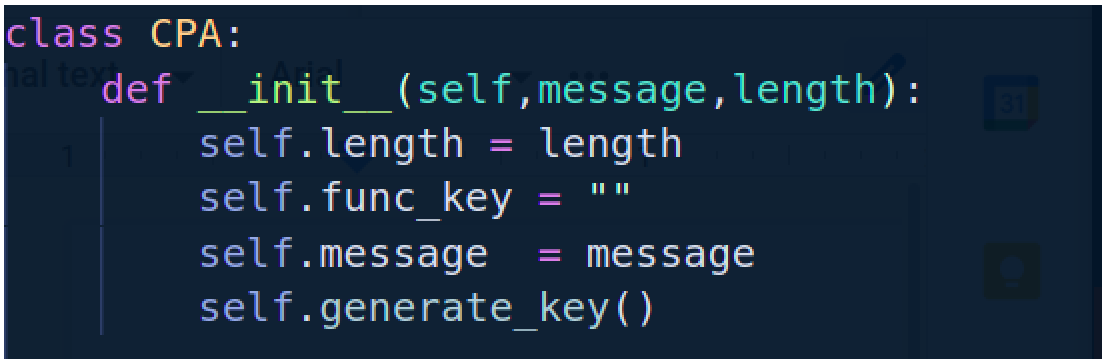
# CPA SECURE ENCRYPTION SCHEME CODE DOCUMENTATION

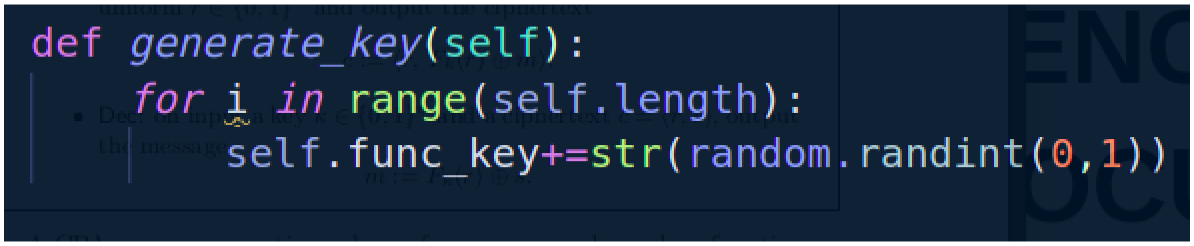
The class **CPA** is created to store all relevant functions/variables required in a CPA secure encryption scheme.



```python
class CPA:
    def __init__(self,message,length):
        self.length = length
        self.func_key = ""
        self.message  = message
        self.generate_key()
```

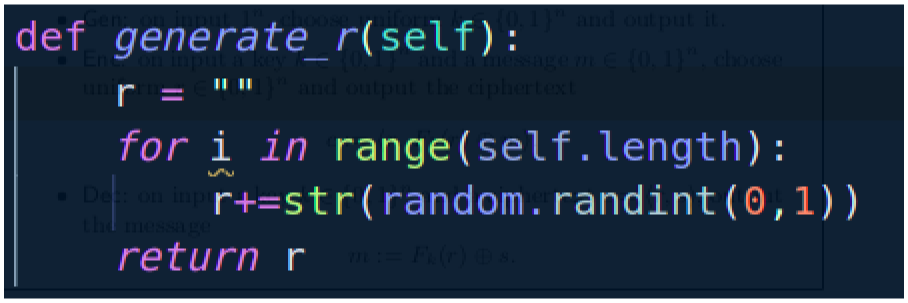
In init all the parameters passed to the class PRF are initialized,the parameters are as follows:

**length :** This variable stores the value of the length of the message(in bits representation)
**func_key**: This variable stores the value of the key generated.
**message**: This variable stores the value of the message given in input.

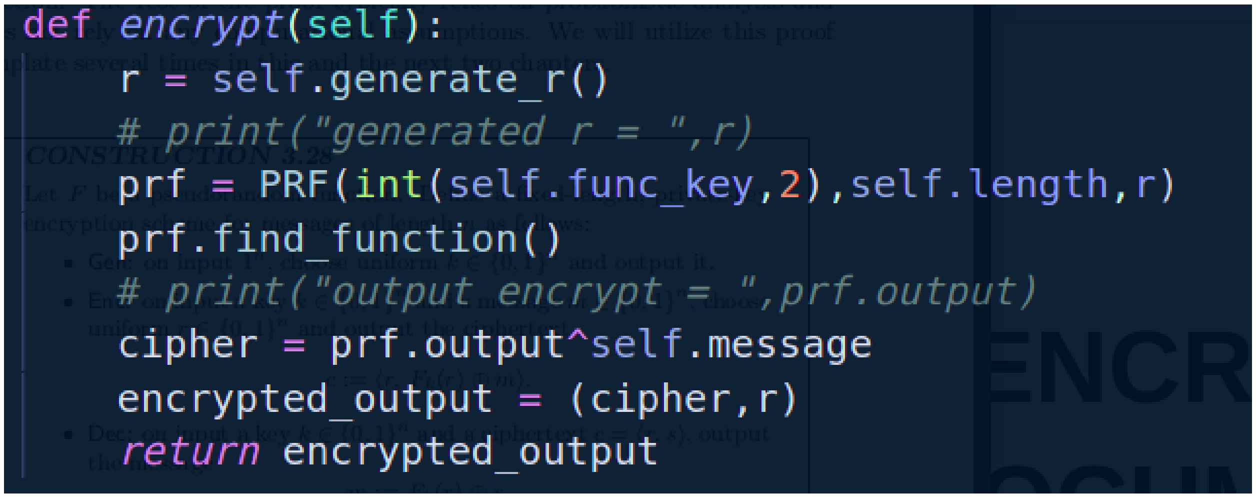


```python
def generate_key(self):
    for i in range(self.length):
        self.func_key+=str(random.randint(0,1))
```

**generate_key :**
This function is used to generate an n bit uniform random string, it is used to generate the key.


```
def generate_r(self):
    r = ""
    for i in range(self.length):
        r+=str(random.randint(0,1))
    return r
```

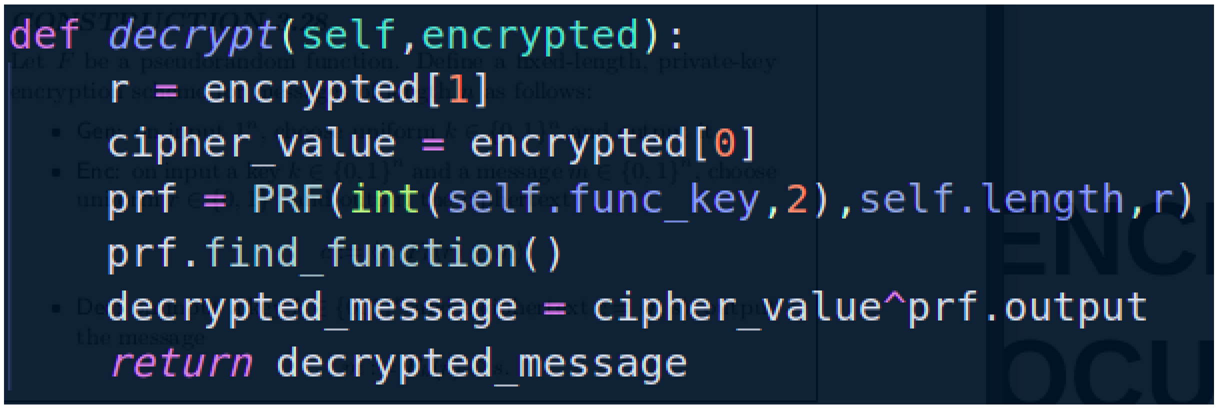
**generate_r:**
This is similar to the generate_key function it generates the r value.


```
def encrypt(self):
    r = self.generate_r()
    # print("generated r = ",r)
    prf = PRF(int(self.func_key,2),self.length,r)
    prf.find_function()
    # print("output encrypt = ",prf.output)
    cipher = prf.output^self.message
    encrypted_output = (cipher,r)
    return encrypted_output
```

**encrypt** function is used for encrypting the message here **prf** is the pseudorandom function implemented

Now we input the generated key as a function key and r value as the input value and the final cipher value is obtained by XORing the output value and message.

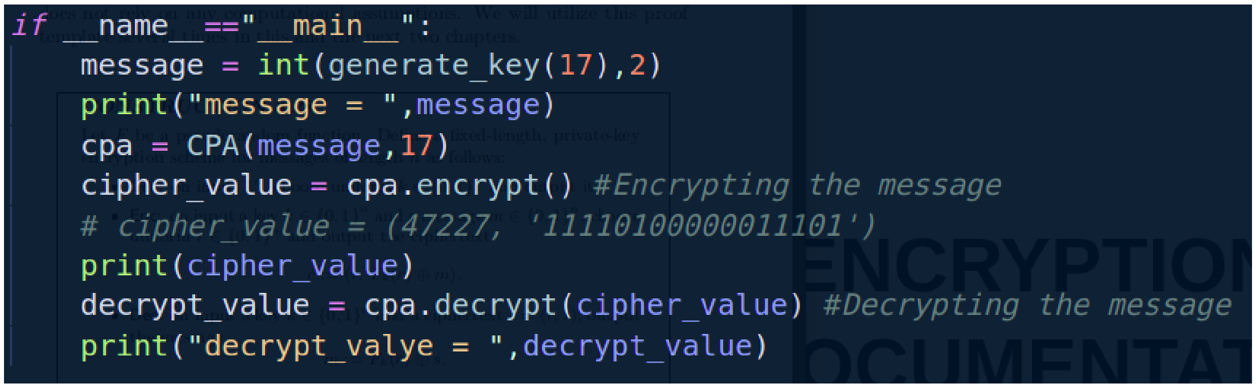
The final encrypted output will be a tuple consisting of cipher and r value.



```python
def decrypt(self,encrypted):
    r = encrypted[1]
    cipher_value = encrypted[0]
    prf = PRF(int(self.func_key,2),self.length,r)
    prf.find_function()
    decrypted_message = cipher_value^prf.output
    return decrypted_message
```

The **decrypt** function is used to decrypt the message from the give tuple consisting of cipher and r value,
When the tuple and r value as given in input, using the pseudorandom function is computed using key and given r value.

The given cipher value is XORed with the output of the prf and the original message is generated.



```python
if __name__=="__main__":
    message = int(generate_key(17),2)
    print("message = ",message)
    cpa = CPA(message,17)
    cipher_value = cpa.encrypt() #Encrypting the message
    # cipher_value = (47227, '11110100000011101')
    print(cipher_value)
    decrypt_value = cpa.decrypt(cipher_value) #Decrypting the message
    print("decrypt_valye = ",decrypt_value)
```

This is the part where the CPA class is initialized and **encrypt** and **decrypt** functions are called.