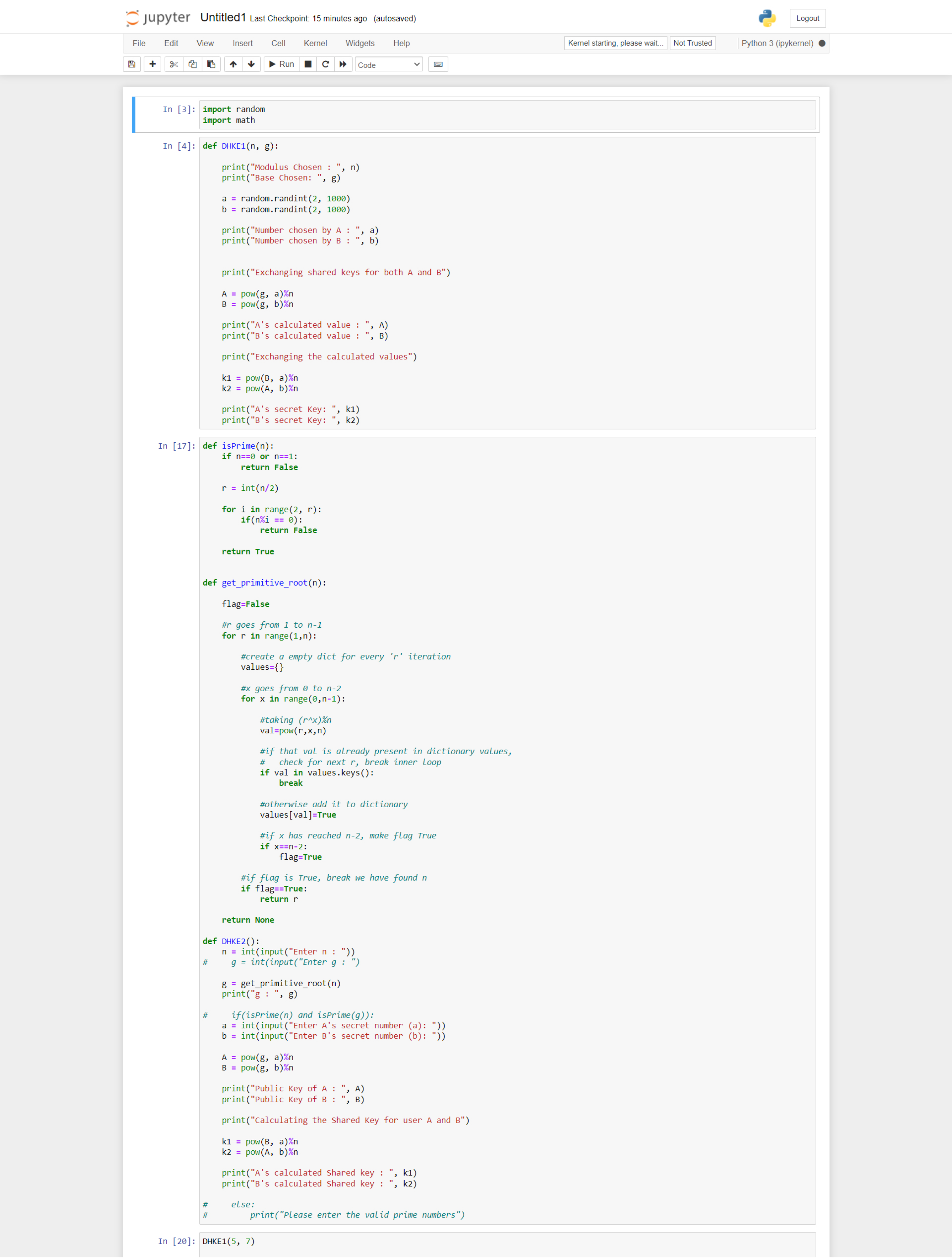

In [3]:
```python
import random
import math
```

In [4]:
```python
def DHKE1(n, g):

    print("Modulus Chosen : ", n)
    print("Base Chosen: ", g)

    a = random.randint(2, 1000)
    b = random.randint(2, 1000)

    print("Number chosen by A : ", a)
    print("Number chosen by B : ", b)


    print("Exchanging shared keys for both A and B")

    A = pow(g, a)%n
    B = pow(g, b)%n

    print("A's calculated value : ", A)
    print("B's calculated value : ", B)

    print("Exchanging the calculated values")

    k1 = pow(B, a)%n
    k2 = pow(A, b)%n

    print("A's secret Key: ", k1)
    print("B's secret Key: ", k2)
```

In [17]:
```python
def isPrime(n):
    if n==0 or n==1:
        return False

    r = int(n/2)

    for i in range(2, r):
        if(n%i == 0):
            return False

    return True


def get_primitive_root(n):

    flag=False

    #r goes from 1 to n-1
    for r in range(1,n):

        #create a empty dict for every 'r' iteration
        values={}

        #x goes from 0 to n-2
        for x in range(0,n-1):

            #taking (r^x)%n
            val=pow(r,x,n)

            #if that val is already present in dictionary values,
            #   check for next r, break inner loop
            if val in values.keys():
                break

            #otherwise add it to dictionary
            values[val]=True

            #if x has reached n-2, make flag True
            if x==n-2:
                flag=True

        #if flag is True, break we have found n
        if flag==True:
            return r

    return None

def DHKE2():
    n = int(input("Enter n : "))
#     g = int(input("Enter g : ")

    g = get_primitive_root(n)
    print("g : ", g)

#     if(isPrime(n) and isPrime(g)):
    a = int(input("Enter A's secret number (a): "))
    b = int(input("Enter B's secret number (b): "))

    A = pow(g, a)%n
    B = pow(g, b)%n

    print("Public Key of A : ", A)
    print("Public Key of B : ", B)

    print("Calculating the Shared Key for user A and B")

    k1 = pow(B, a)%n
    k2 = pow(A, b)%n

    print("A's calculated Shared key : ", k1)
    print("B's calculated Shared key : ", k2)

#     else:
#         print("Please enter the valid prime numbers")
```

In [20]:
```python
DHKE1(5, 7)
```

```
print("------------------------------")

DHKE2()
```

```
Modulus Chosen :  5
Base Chosen:  7
Number chosen by A :  633
Number chosen by B :  758
Exchanging shared keys for both A and B
A's calculated value :  2
B's calculated value :  4
Exchanging the calculated values
A's secret Key:  4
B's secret Key:  4
------------------------------
Enter n : 5
g :  2
Enter A's secret number (a): 633
Enter B's secret number (b): 758
Public Key of A :  2
Public Key of B :  4
Calculating the Shared Key for user A and B
A's calculated Shared key :  4
B's calculated Shared key :  4
```

In [ ]:

In [ ]: