1. For what 'x' is 77x mod 101 is equal to 1? Write a Python program using range ().

```
def moduloinverse(a,m):
   for x in range(0,m,1):
      if((a*x)%m)==1:
         return x
   else:
      return -1
x = moduloinverse(77,101)
if x==-1 :
   print("There is no Possibility of X value ")
else:
   print("The value of x : ",x)
```

2. Computer 2100 mod 101 using Python.

```
print(2100 % 101)
```

3. Find the smallest integer ‘k’ such that 2k ≡ 1 mod 101. Write a Python program for this mathematical equation.

```
def moduloinverse(a,m):
   for k in range(0,m,1):
      if((a*k)%m)==1:
         return k
   else:
      return -1
k = moduloinverse(2,101)
if k==-1 :
   print("There is no Possibility of k value ")
else:
   print("The value of k : ",k)
```

4. Find the GCD of 123456789 and 987654321 using Euclid Algorithm with a Python script.

```
def gcd(mini,maxi):
   if(maxi%mini == 0):
      return mini
   else:
      return gcd(maxi%mini,mini)
print(gcd(123456789,987654321))
```

5. Write a Python program to check whether a given string contains a capital letter, a lowercase letter, a number, and a minimum length using lambda

```
import re
string = str(input("Enter a String: "))
ans = lambda s : print("Valid String") if(re.search('[A-Z]',string) and
re.search('[a-z]',string) and re.search('\d',string) and len(string)>=5)
else print("Invalid String")
ans(string)
```

6. Write a Python program to calculate the average value of the numbers in a given tuple using the lambda function.
Original Tuple:
((10, 10, 10), (30, 45, 56), (81, 80, 39), (1, 2, 3))

```
t=((10, 10, 10), (30, 45, 56), (81, 80, 39), (1, 2, 3))
average = lambda t : sum(t)/len(t)
for i in t:
    print(average(i))
```

7. Write a Python function that can multiply an input with an unknown number using the lambda function inside a standard function. For example, if the standard function takes an input of 3 and the lambda function takes an input of 11, the final result should be 33. Check if your function works correctly by calling it for some suitable inputs.

```
def function(a):
  return lambda no : a*no
print(function(3)(11))
```

8. Write a Python program to create a Reverse cipher using a function. The input should be Plain text passed as one parameter. The output is the cipher text, a complete reverse of the Plain text.

```
string = str(input())
ciphertext  = lambda st : st[::-1]
print("Cipher Text :",ciphertext(string))
```

9. Implement an XOR Cipher using a Python function.

```
plaintext = str(input())
key = str(input())
def cyphertext(plaintext,key):
   ct=""
   for i in range(0,len(plaintext)):
      ct = ct + chr(ord(plaintext[i])^ord(key[i%len(key)]))
   return ct
print("Cipher Text of given plain text is ",cyphertext(plaintext,key))
```

10. The order of an element modulo N is the least positive integer ‘k’ such that ak ≡ 1 mod N. Find the order of 3, 4, and 5 modulo 101 using a Python function.

```
def ordermodulo(n,m):
   for i in range(1,m):
      if(n**i)%m == 1:
         return i
   return -1
print(ordermodulo(3,101))
print(ordermodulo(4,101))
print(ordermodulo(5,101))
```

11. Implement the RSA Encryption Algorithm using Python PIP.

```
import rsa

public_key,private_key = rsa.newkeys(1024)
#print(public_key,private_key)

message = "Hello my Name id NG 555"

enc = rsa.encrypt(message.encode(),public_key)
```

```
print("The encrypted message is " ,enc)
#enc = open("enc.message","rb").read()
clear_message= rsa.decrypt(enc,private_key)
print("The decrypted message is " ,clear_message.decode())
```

12. Write a Python program to encode the following string: “Advanced Diploma in Cyber Security”.

```
import base64

string = "Advanced Diploma in Cyber Security"
encoded_string = base64.b64encode(string.encode())

print("Encoded String: ", encoded_string.decode())
```

13. Write a Python program to perform the Substitution cipher for the following Plain text: “Sona College of Technology, Salem”.

```
def substitution_cipher(plaintext, key):
    ciphertext = ""
    for char in plaintext:
        if char.isalpha():
            shift = ord(key[ord(char.upper())-65])
            if char.isupper():
                ciphertext += chr(shift)
            else:
                ciphertext += chr(shift + 32)
        else:
            ciphertext += char
    return ciphertext

plaintext = "Sona College of Technology, Salem"
key = "QWERTYUIOPASDFGHJKLZXCVBNM"
```

```
ciphertext = substitution_cipher(plaintext, key)
print("Ciphertext: ", ciphertext)
```

14. Write a Python program to determine the strength of a Password depending on the constraints matching the regular expression.

```
import re

def pass_verifier(password):
  if (len(password)<8):
    return("The password must be at least 8 characters long.")
  elif not re.search("[a-z]", password):
    return("The password must have at least one lowercase letter.")
  elif not re.search("[A-Z]", password):
    return("The password must have at least one uppercase letter.")
  elif not re.search("[0-9]", password):
    return("The password must have at least one number.")
  elif not re.search("[^a-zA-Z0-9]", password):
    return("The password must have at least one special character.")
  else:
    return("The password is strong.")
password = input("Enter the password: ")
print(pass_verifier(password))
```

15. Generate a Fake address with Latitude and Longitude using a Python package.

```
from faker import Faker

fake = Faker()
```

```
address = fake.address()
latitude = fake.latitude()
longitude = fake.longitude()

print("Address: ", address)
print("Latitude: ", latitude)
print("Longitude: ", longitude)
```