Name: Muhammad Rafiq

Seat No: B17101061

Course: Network Security & Cryptography (NS'21 Lab)

Assignment : Assignment # 1

Section: A

Modulus Algorithm

In [1]:
```
def modulo(a, m)->int:
    R = abs(a) % m
    if a>= 0:
        R =R
    elif a < 0 and R != 0:
        R = m - R
    elif a < 0 and R == 0:
        R = 0
    return R
```

1- Write a program to encrypt and decrypt the communication or message between the end-users using Shift, Atbash and Rot13 cipher techniques.

Shift (Mono Alphabetical Techniqueus)

In [6]:
```
''' A = 0 ... Z = 25'''
def my_encrypt(my_string,K,m) -> str:
    enc_string = ""
    my_string = my_string.upper()
    for character in my_string:
        enc_character = modulo((ord(character)-65 + K), m)
        enc_string += chr(enc_character+65)
    return enc_string

def my_decrypt(my_string,K,m)->str:
    dec_string = ""
    for character in my_string:
        dec_character = modulo((ord(character) - 65 - K), m)
        dec_string += chr(dec_character + 65)
    return dec_string
def shift_cipher(choice):
    if choice == 'e':
        my_input = input("Enter the text to be encrypted: ")
        K = int(input("Enter the key in an integer: "))
        enc_string = my_encrypt(my_input.replace(" ", "").upper(),K % 26,26)
        print("Encrypted String is: " + enc_string)
    elif choice == 'd':
        key_choice = input("print y if key is known otherwise some other character:")
        enc_string = input("Enter the text to be decrypted: ")
        if key_choice == "y":
            K = int(input("Enter the key in an integer: "))
            my_string = my_decrypt(enc_string.replace(" ", "").upper(), K % 26,26)
            print("Original String after decryption is: " + my_string)
        else:
            word = {}
            for i in enc_string.replace(" ", "").upper():
                if i in word.keys():
                    word[i] += 1
                else:
                    word[i] = 1
            dic = {k: v for k, v in sorted(word.items(), key=lambda item: item[1], reverse=T
rue)}
            l = list(dic.keys())
            for i in "AEIOU":
                my_string = my_decrypt(enc_string.replace(" ", "").upper(), abs(ord(i)-ord(l
[0])) ,26)
                print("Original String after decryption is: " + my_string)
    else:
        my_input = input("Enter the text to be encrypted: ")
        K = int(input("Enter the key in an integer: "))
        enc_string = my_encrypt(my_input.replace(" ", "").upper(),K % 26,26)
        print("Encrypted String is: " + enc_string)
        my_string = my_decrypt(enc_string, K % 26,26)
        print("Original String after decryption is: " + my_string)
```

ROT 13

In [10]:
```
''' A = 0 ... Z = 25'''
'''But Key would always be 13 in  this algorithm'''
def my_rot_encrypt(my_string,m) -> str:
    enc_string = ""
    my_string = my_string.upper()
    for character in my_string:
        enc_character = modulo((ord(character)-65 + 13), m)
        enc_string += chr(enc_character+65)
    return enc_string

def my_rot_decrypt(my_string,m)->str:
    dec_string = ""
    my_string = my_string.upper()
    for character in my_string:
        dec_character = modulo((ord(character) - 65 - 13), m)
        dec_string += chr(dec_character + 65)
    return dec_string
def rot13_cipher(choice):
    if choice == 'e':
        my_input = input("Enter the text to be encrypted: ")
        enc_string = my_encrypt(my_input.replace(" ", "").upper(),26)
    elif choice == 'd':
        enc_string = input("Enter the text to be decrypted: ")
        my_string = my_rot_decrypt(enc_string.replace(" ", "").upper(),26)
        print("Original String after decryption is: " + my_string)
    else:
        my_input = input("Enter the text to be encrypted: ")
        enc_string = my_rot_encrypt(my_input.replace(" ", "").upper(),26)
        print("Encrypted String is: " + enc_string)
        my_string = my_decrypt(enc_string,26)
        print("Original String after decryption is: " + my_string)
```

ATBASH

In [4]:
```
def ATBASH_encryption(my_string,m)->str:
    my_string = my_string.upper()
    enc_string = ""
    for character in my_string:
        enc_character = modulo(-(ord(character) - 65 + 1), m)
        enc_string += chr(enc_character + 65)
    return enc_string
def ATBASH_cipher():
    input_string = input("Enter String tp be encrypted using ATBASH Algorithm: ")
    enc_string = ATBASH_encryption(input_string, 26)
    print(enc_string)
    print(ATBASH_encryption(enc_string, 26))
```

2- Kindly decode the text "yfnzjpflivogvizvetvnzkyjfcmzexwzijktzgyvikvok" using your shift cipher program.

In [7]:
```
shift_cipher("d")    #d for decryption only
```

```
print y if key is known otherwise some other character:n
Enter the text to be decrypted: yfnzjpflivogvizvetvnzkyjfcmzexwzijktzgyvikvok
Original String after decryption is: ZGOAKQGMJWPHWJAWFUWOALZKGDNAFYXAJKLUAHZWJLWPL
Original String after decryption is: DKSEOQKQNATLANEAJYASEPDOKHREJCBENOPYELDANPATP
Original String after decryption is: HOWISYOUREXPERIENCEWITHSOLVINGFIRSTCIPHERTEXT
Original String after decryption is: NUCOYEUAXKDVKXOKTIKCOZNYURBOTMLOXYZIOVNKXZKDZ
Original String after decryption is: TAIUEKAGDQJBQDUQZOQIUFTEAXHUZSRUDEFOUBTQDFQJF
```

3- Kindly decode the text "guvf vf gbb rnfl gb whqtr" using ROT13

In [11]:
```
rot13_cipher("d")    #d for decryption only
```

```
Enter the text to be decrypted: guvf vf gbb rnfl gb whqtr
Original String after decryption is: THISISTOOEASYTOJUDGE
```

4- Kindly Encrypt the text "Assignments are too much to solve in a week", using key 917.

In [14]:
```
shift_cipher("e")    #e for encryption only
```

```
Enter the text to be encrypted: Assignments are too much to solve in a week
Enter the key in an integer: 917
Encrypted String is: HZZPNUTLUAZHYLAVVTBJOAVZVSCLPUHDLLR
```