→ Samyukta
→ SOIS

Q1)

It is exponential key-exchange, it uses numbers raised to specific powers to produce decryption keys on the basis of components that are never directly transmitted, making it mathamatically overwhelming

It is used to exchange the secret key between sender and reciever.

eg → credit card transaction

Q2) 16

Q3) Encryption

$E_i = (P_i + K_i) \bmod 26$

Decryption

$D_i = (E_i - K_i + 26) \bmod 26$

Q4)

```
x = lambda a, b: a*b
print(x(5,6))
```

Q5) Eg - let Alice and Bob be two end users and mutually agree on 2 positive whole numbers p, q.

PAGE No.
DATE / /

So p is a prime number and q a generator of F P. The generator q is a number that, when raised to positive whole-number powers less than p, never produces the same result for any two such whole numbers.

Alice	Bob
public keys = P, G	public keys = P, G
Private key = a	Private key = b
key generated $x = G^a \bmod P$	key generated $y = G^b \bmod P$
Exchange	
recieved = y	recieved = x
generated $k_a = y^a \bmod P$	generated $k_b = x^b \bmod P$
$k_a = k_b$	

Q) Vigenère cipher is a method of encrypting alphabetic text. It uses simple polyalphabetic substitution. The encryption is done using vigenere table.

eg → text → HELLOTHERE
key → SAM

keyword 'SAM' generates SAMSAMSAMSA

Encryption

The plaintext (P) and key (K) are added modulo 26

$E_i = (P_i + K_i) \bmod 26$

Decryption

$D_i = (E_i - K_i + 26) \bmod 26$

Q) string = "HELLOTHERE"
key = "SAM"

```
def generatekey(string, key):
    key = list(key)
    if len(string) == len(key):
        return(key)
    else:
        for i in range(len(string) - len(key)):
            key.append(key[i % len(key)])
    return("".join(key))
```

```
def encrypt_cipherText (string, key):
    cipher_text = []
    for i in range (len(string))
        x = ((ord(string[i]) + ord (key[i])) %26 +
                                        ord('A')
        cipher_text.append (chr(x))
    return("".join(cipher_text))

key = generate key(string, keyword)

print ("Original:", string)
print ("keyword:", keyword)
cipher_text = encrypt_ciperText(string, key)
print("Cipher text:", cipher_text)
```

output:

original: HELLOTHERE
keyword: SAM
ciphertext: ZEXDOFZEDW