Name :- Pooja Bisht

Course :- BCA - 'B' (6th Sem)

Subject :- Information Security lab

Paper Type :- Regular (End-term Practical)

Roll No :- 1121101 (20)

Email Address :- poojabisht031@gmail.com.

MCQ

① (a) Symmetric key encryption with receiver public key.

②. (c) spyware

③ (c) An authentication of an electronic Record.

④ (d) None

⑤ (a) Only on alphanumeric

⑥ (c) All

⑦ (a) hash value

⑧ (b) The identity of the character is changed while its position remains unchanged

⑨ (b) to make even no. of letters.

⑩ (a) Total length of word.

Pooja.

(3) WAP for the encryption & decryption of the vignere cipher on the input. plaintext = "Cryptography" with a key = "Monarchy".

```
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 encryption(string, key):
    encrypt_text = []
    for i in range(len(string)):
        x = (ord(string[i]) + ord(key[i]))
                          % 26
        x += ord('A')
```

Pooja

```
        encrypt-text.append (char(x))
    return ("" . join(encrypt-text))

def decryption(encrypt-text, key):
    orig-text = []
    for i in range(len(encrypt-text)):
        x = (ord(encrypt-text[i]) -
                ord(key[i]) + 26) % 26
        x += ord ("A")
        orig-text.append (char(x))
    return ("" . join(orig-text))

if __name__ == "__main__":
    string = "Cryptography"
    keyword = "Monarchy"
    key = generatekey (string, keyword)
    encrypt-text = encryption(string, key)
    print ("encrypted message:", encrypt-
                                text)
print ("Decrypted message:", decryption
                            (encrypt-text,
                             key)).
```

Pooja

Descriptive Answers :-

④. WAP to implement OTP(One Time Password)

```
import math, random
def generateOTP() :
    ~~string = "0123456789"~~
    digits = "0123456789"
    OTP = ""
    for i in range (4):
        OTP+ = digits[math.floor(random.
                                  random()
                                  * 10)]
    return OTP
if __name__ == "__main__":
    print("OTP of 4 digits :", generateOTP())
```

Pooja

⑤ WAP to implement encryption and decryption using ceaser cipher on the input plaintext = "Attack from North".

```
print ("Perform Encryption:")
def encrypt(text, s):
    result = " "
    for i in range (len(text)):
        char = text[i]
        if (char.isupper()):
            result = result + chr((ord(char)
                        +s-65)%26+65)
        else:
            result = result + chr((ord(char)
                        +s-97)%26+97)
    return result
text = "Attack from North"
 s = 3
print ("Plain text:", text)
print ("Encrypted text:", encrypt(text, s))
print ("Perform Decryption:")
```

Pooja

```
def decrypt(text, s):
  result = " "
  for i in range(len(text)):
    char = text[i]
    if(char.isupper()):
      result = result + chr((ord(char)
                 - s - 65) % 26 + 65)
    else:
      result = result + chr((ord(char)-
                 s - 97) % 26 + 97)
  return result
text = encrypt(text, s)
  s = 3
  print("Decrypted text:", decrypt(
                      text, s))
```

Pooja