Cryptography

Section 5

Created by/ Rehab Mohamed & Hanaa Mansour

Multiplicative Cipher

- While using Caesar cipher technique, encrypting and decrypting symbols involves converting the values into numbers with a simple basic procedure of addition or subtraction.
- If multiplication is used to convert to cipher text, it is called a **wrap-around** situation. Consider the letters and the associated numbers to be used as shown below.

$$C = E(P) = (P * K) \bmod n$$ ← قانون التشفير

$$P = D(C) = (C * k^{-1}) \bmod n$$ ← قانون فك التشفير

Note: The Greatest Common Divisor (GCD) between the key and n should equal 1.

Example: Decrypt the Ciphertext "GKSXKP" using multiplicative cipher with key=9 ~~~~~~~ n=26

Sol)….

$$P = D(C) = (C * \mathrm{k}^{-1}) \bmod n$$ ◄------ قانون فك التشفير

K	1	3	5	7	9	11	15	17	19	21	23	25
K^{-1}	1	9	21	15	3	19	7	23	11	5	17	25

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Alphabet	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z

GKSXKP

Ciphertext	Decryption $p=(c\times k^{-1})\bmod 26$	plaintext
G=6	C=(6×3)mod26	18=S
K=10	C=(10×3)mod26	4=E
S=18	C=(18×3)mod26	2=C
X=23	C=(23×3)mod26	17=R
K=10	C=(10×3)mod26	4=E
P=15	C=(15×3)mod26	19=T
		SECRET

Decryption using Multiplicative Cipher

```
def decrypt(cipher):
    result = ''
    for c in cipher:
        if (c != ' '):
            if (c.isupper()):
                # Encrypt uppercase characters
                s = chr(((ord(c) - 65) * 3) % 26 + 65)
            else:
                # Encrypt lowercase characters
                s = chr(((ord(c) - 97) * 3) % 26 + 97)
        else:
            s = ' '
        result += s
    return result
```

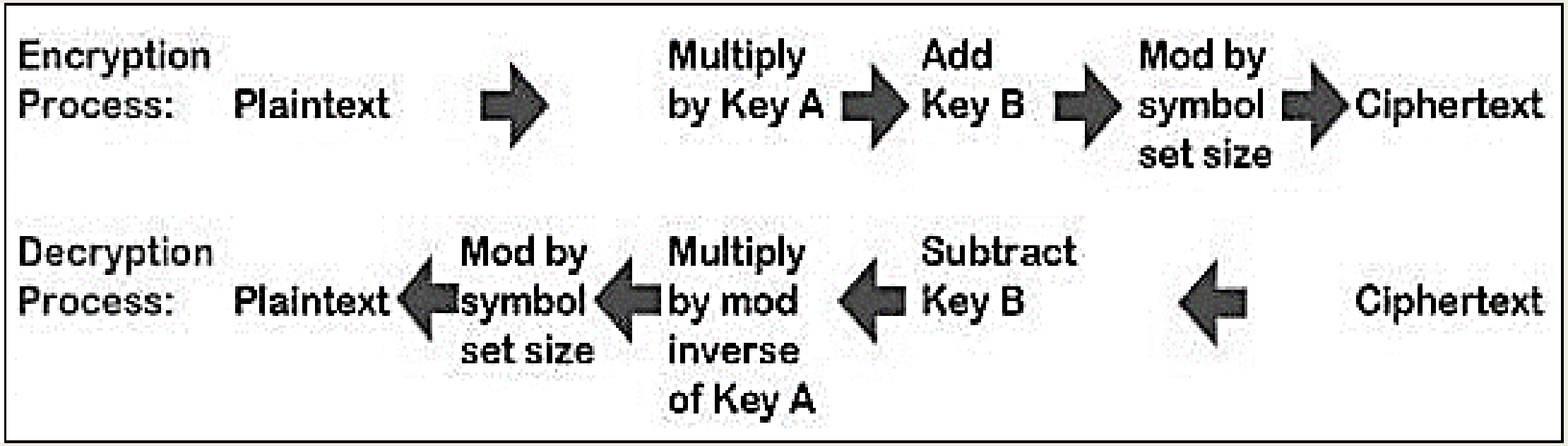
```
message = "Gksxkp"
plaintext = decrypt(message)
print("The decrypted text is: " + plaintext)
```

Output:

```
The decrypted text is: Secret
```

Affine Cipher

- Affine Cipher is the combination of Multiplicative Cipher and Caesar Cipher algorithm. The basic implementation of affine cipher is as shown in the image below:



0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

- **Affine Cipher**

Encrypt the message "Go" using affine cipher given the key (7:a, 2:b).

C = (ax + b) mod 26

E(G) = 44 mod 26 = 18 -> S

E(O) = 100 mod 26 = 22 -> W

Go -> SW

Encryption using Affine Cipher

```
k1 = 7
k2 = 2
plaintext = "hello world"
result = ''
for c in plaintext:
    if (c != ' '):
        if (c.isupper()):
            # Encrypt uppercase characters
            s = chr((((ord(c) - 65) * k1) + k2) % 26 + 65)
        else:
            # Encrypt lowercase characters
            s = chr((((ord(c) - 97) * k1) + k2) % 26 + 97)
    else:
        s = ' '
    result += s
print(result)
```

Output:

```
zebbw awrbx
```

Decryption using Affine Cipher

```
k1 = 15
k2 = 2
ciphertext = "zebbw awrbx"
result = ''
for c in ciphertext:
    if (c != ' '):
        if (c.isupper()):
            # Encrypt uppercase characters
            s = chr((((ord(c) - 65) - k2) * k1) % 26 + 65)
        else:
            # Encrypt lowercase characters
            s = chr((((ord(c) - 97) - k2) * k1) % 26 + 97)
    else:
        s = ' '
    result += s
print(result)
```

Output:

```
hello world
```


Thank
you