Experiment No. 3

Aim: To implement Additive Cipher (Caesar Cipher).

Theory: Additive Cipher also known as **Caesar Cipher** is an example of **Substitution Cipher** in which each character of Plain text is replaced by some other character present in its domain using key in the range of size of domain, in Additive Cipher size of domain is equal to total alphabets i.e. 26 (count starts from 0 but we exclude 0 because it gives same cipher text as plain text).

Additive Cipher key range**:** [1,25]

Encipher Function: $E(P,k) = (P+k)(mod26)$ – {Where, P is the numeric value of alphabet in Plain text and k is the key agreed by sender and receiver}.

Decipher Function: $D(C,k) = (C-k)(mod26)$ – {Where, C is the numeric value of alphabet in Cipher text and k is the same key as in Encipher Function}.

We mod by 26 because the numeric values of our domain can never exceed 25. Cipher text is always in capital case.

Alphabets and their numeric values: Values are same for both Cipher text and Plain text.

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
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
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

E.g. Plain text: Python is awesome, Key = 5

$$P -> 15 -> E(15,5) = (15+5)(mod26) = 20 -> U$$

$$y -> 25 -> E(25,5) = (24+5)(mod26) = 3 -> D$$

Similarly, doing for each this for each alphabet in Plain text we get Cipher text as: UDYMTSNXFBJXTRJ. We removed spaces in Plain text we can also keep it as it is.

Cipher text: UDYMTSNXFBJXTRJ, Key = 5

$$U \rightarrow 20 \rightarrow D(20,5) = (20-5)(mod26) = 15 \rightarrow p$$

$$D \rightarrow 3 \rightarrow D(3,5) = (3-5)(mod26) = 24 \rightarrow y$$

Similarly, doing this for each alphabet in Cipher text we get Plain text as: pythonisawesome. It can be seen that it is same as above Plain text (spaces removed).

By above example it is clear that $E(P,k) \Leftrightarrow D^{-1}(P,k)$ and $D(C,k) \Leftrightarrow E^{-1}(C,k)$. Since, both function are invertible they form one-to-one correspondence with each other.

Implementation:

```
import numpy as np
import string

class AdditiveCipher:

    def __init__(self, key = np.random.randint(1,26)):
        assert 1<=key<= 25, "Key should be in between [1,25]"
        self.domain = 26
        self.key = key
        self.map = {key:value for key,value in zip(list(string.ascii_uppercase),range(0,26))}
        self.reverse_map = {value:key for key,value in self.map.items()}

    def encode(self, plain_txt):
        plain_txt = plain_txt.replace(" ","")
        assert plain_txt.isalpha() or plain_txt, "Plain text should only contain alphabetic character"
        cipher_txt = ""

        for txt in plain_txt:
            cipher_txt += self.reverse_map[(self.map[txt.upper()]+self.key)%self.domain]

        return cipher_txt

    def decode(self, cipher_txt):
        assert cipher_txt.isupper() and cipher_txt.isalpha(), "Cipher text should only contain uppercase alphabetic character"
```

```
        plain_txt = ""

        for txt in cipher_txt:
            plain_txt += self.reverse_map[(self.map[txt]-self.key)%self.domain].lower()

        return plain_txt

additive = AdditiveCipher(int(input("Enter key in between [1,26] for encipherment and decipherment: ")))
print()

z = additive.encode(input("Enter a text [a-z]|[A-Z] to encode it:"))
y = additive.decode(z)
print(f"Plain text = {y}, Cipher text = {z}\n\n")

z = additive.decode(input("Enter a text [A-Z] to decode it:"))
y = additive.encode(z)
print(f"Plain text = {z}, Cipher text = {y}")
```

Output:

```
Enter key in between [1,26] for encipherment and decipherment: 6

Enter a text [a-z]|[A-Z] to encode it:Its me Mario
Plain text = itsmemario, Cipher text = OZYSKSGXOU


Enter a text [A-Z] to decode it:OZYSKSGXOU
Plain text = itsmemario, Cipher text = OZYSKSGXOU

Enter key in between [1,26] for encipherment and decipherment: 25

Enter a text [a-z]|[A-Z] to encode it:ZaAWurdo
Plain text = zaawurdo, Cipher text = YZZVTQCN


Enter a text [A-Z] to decode it:YZZVTQCN
Plain text = zaawurdo, Cipher text = YZZVTQCN


Enter key in between [1,26] for encipherment and decipherment: 10

Enter a text [a-z]|[A-Z] to encode it:Area of accuracy between Frequency and Time remain constant in Wavelet transform
Plain text = areaofaccuracybetweenfrequencyandtimeremainconstantinwavelettransform, Cipher text = KBOKYPKMMEBKMILODGOOXPBOAEOXM
IKXNDSWOBOWKSXMYXCDKXDSXGKFOVODDBKXCPYBW


Enter a text [A-Z] to decode it:KBOKYPKMMEBKMILODGOOXPBOAEOXMIKXNDSWOBOWKSXMYXCDKXDSXGKFOVODDBKXCPYBW
Plain text = areaofaccuracybetweenfrequencyandtimeremainconstantinwavelettransform, Cipher text = KBOKYPKMMEBKMILODGOOXPBOAEOXM
IKXNDSWOBOWKSXMYXCDKXDSXGKFOVODDBKXCPYBW
```