Caesar Cipher

Caesar cipher uses a substitution method which uses keys to shift the alphabetical letters index by the value of the key. Say the key = 3, then all 26 letters in the alphabet will be shifted 3 indexes.

plain: 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

cipher: 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

It is said this is what Julius Caesar used himself to communicate with his fellow comrades. We can assign each character a numerical value where 0 = A, 1 = B, and so on. Below is a table where each character is assigned a value:

a	b	c	d	e	f	g	h	i	j	k	l	m
0	1	2	3	4	5	6	7	8	9	10	11	12

n	o	p	q	r	s	t	u	v	w	x	y	z
13	14	15	16	17	18	19	20	21	22	23	24	25

Even though there are 26 letters in the alphabet, we can see that max value is 25 because our first index value is 0.

To encrypt text we use the following line of code below:

```
if ord(char) > 96 :
    res += chr((ord(char) + key - 97) % 26 + 97)
else:
    res += chr((ord(char)+ key - 65) % 26 + 65)
```

And to decrypt it we simply subtract the key which becomes:

```
if ord(char) > 96 :
    res += chr((ord(char) - key - 97) % 26 + 97)
else:
    res += chr((ord(char) - key - 65) % 65 + 65)
```

When using the cipher, you are asked to give a plain text string which would be a message that the user inputs. In our example for checking the runtime, we used the string hello and the key 3. Our encrypted text (which would be the ciphertext) becomes khoos. To decrypt the text, we can shift the key back a value of three and we would get the plaintext again.

```
Tris-MacBook-Air:desktop tringuyen$ python3 cipher.py
Enter message you would like to encrypt:
hello
Your message is: hello
Enter key:3
Type 'encrypt' for encryption and 'decrypt' for decrytion.
encrypt
Your encrypted message is: khoor
encryption time is:  9.5367431640625e-07 seconds
Tris-MacBook-Air:desktop tringuyen$ python3 cipher.py
Enter message you would like to encrypt:
khoor
Your message is: khoor
Enter key:3
Type 'encrypt' for encryption and 'decrypt' for decrytion.
decrypt
Your decrypted message is: hello
decryption time is:  1.1920928955078125e-06 seconds
Tris-MacBook-Air:desktop tringuyen$
```

```
Tris-MacBook-Air:desktop tringuyen$ python3 cipher.py
Enter message you would like to encrypt:
georgia
Your message is: georgia
Enter key:4
Type 'encrypt' for encryption and 'decrypt' for decrytion.
encrypt
Your encrypted message is: kisvkme
encryption time is:  9.5367431640625e-07 seconds
Tris-MacBook-Air:desktop tringuyen$ python3 cipher.py
Enter message you would like to encrypt:
kisvkme
Your message is: kisvkme
Enter key:4
Type 'encrypt' for encryption and 'decrypt' for decrytion.
decrypt
Your decrypted message is: georgia
decryption time is:  9.5367431640625e-07 seconds
Tris-MacBook-Air:desktop tringuyen$
```

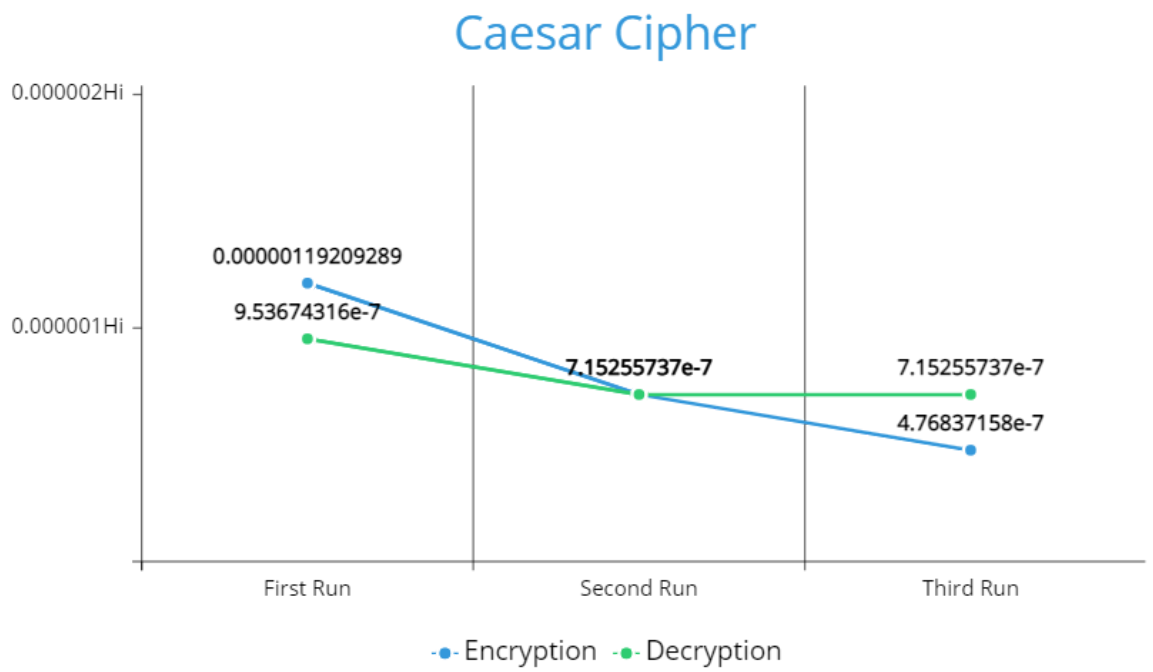
```
Tris-MacBook-Air:desktop tringuyen$ python3 cipher.py
Enter message you would like to encrypt:
Georgia
Your message is: Georgia
Enter key:4
Type 'encrypt' for encryption and 'decrypt' for decrytion.
encrypt
Your encrypted message is: Kisvkme
encryption time is:  1.1920928955078125e-06 seconds
Tris-MacBook-Air:desktop tringuyen$ python3 cipher.py
Enter message you would like to encrypt:
Kisvkme
Your message is: Kisvkme
Enter key:4
Type 'encrypt' for encryption and 'decrypt' for decrytion.
decrypt
Your decrypted message is: Georgia
decryption time is:  1.1920928955078125e-06 seconds
Tris-MacBook-Air:desktop tringuyen$
```

Encrypt Time: 1.1920928955078125e-06 seconds

```
Enter message you would like to encrypt or Enter the number value
ecrypt separate by ',':
Hello
Your message is: Hello
Enter key:3
Type 'encrypt' for encryption and 'decrypt' for decrytion.
encrypt
Your encrypted message is: Khoor
encryption time is:  1.1920928955078125e-06 seconds
```

Decrypt Time: 9.5367431640625e-07 seconds

```
Enter message you would like to encrypt or Enter the number value
ecrypt separate by ',':
Khoor
Your message is: Khoor
Enter key:3
Type 'encrypt' for encryption and 'decrypt' for decrytion.
decrypt
Your decrypted message is: Hello
encryption time is:  9.5367431640625e-07 seconds
```



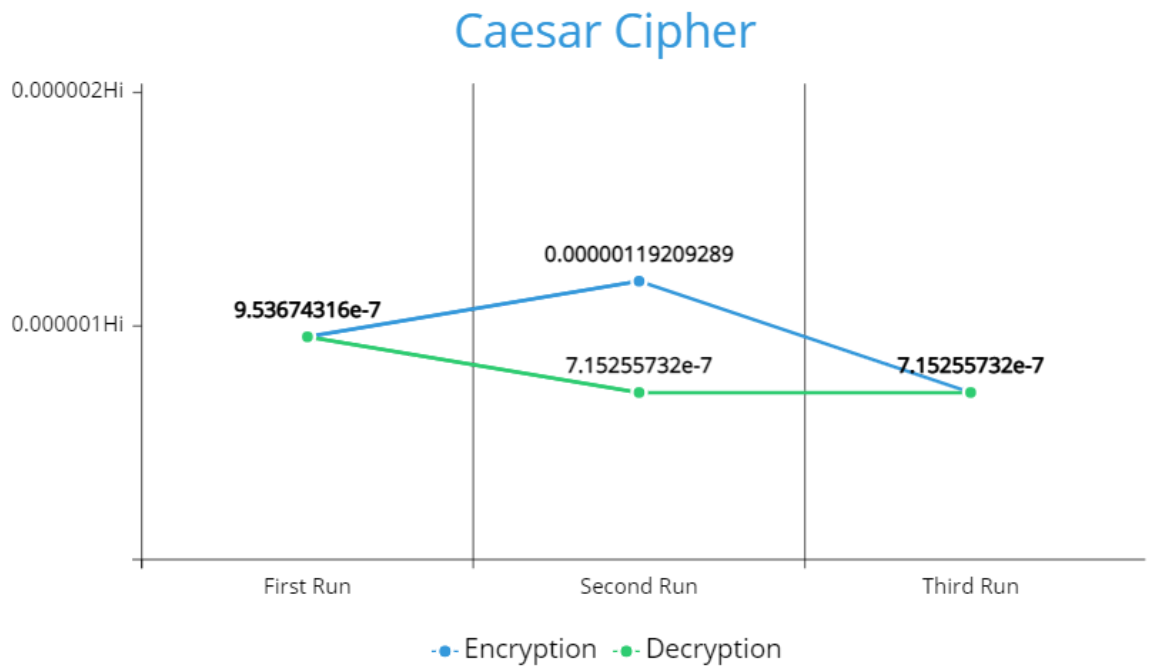
We had some inconsistencies with the encrypt and decrypt times but it stayed in a range (+- 2.38418574e^-7seconds off). The encrypt time was slightly faster than the decrypt time which makes sense because it takes in ambiguous letters and has to convert it and run it through to become plaintext.

Plaintext = mi nombre es jefe

Key = 6

Ciphertext = so tushxk ky pklk

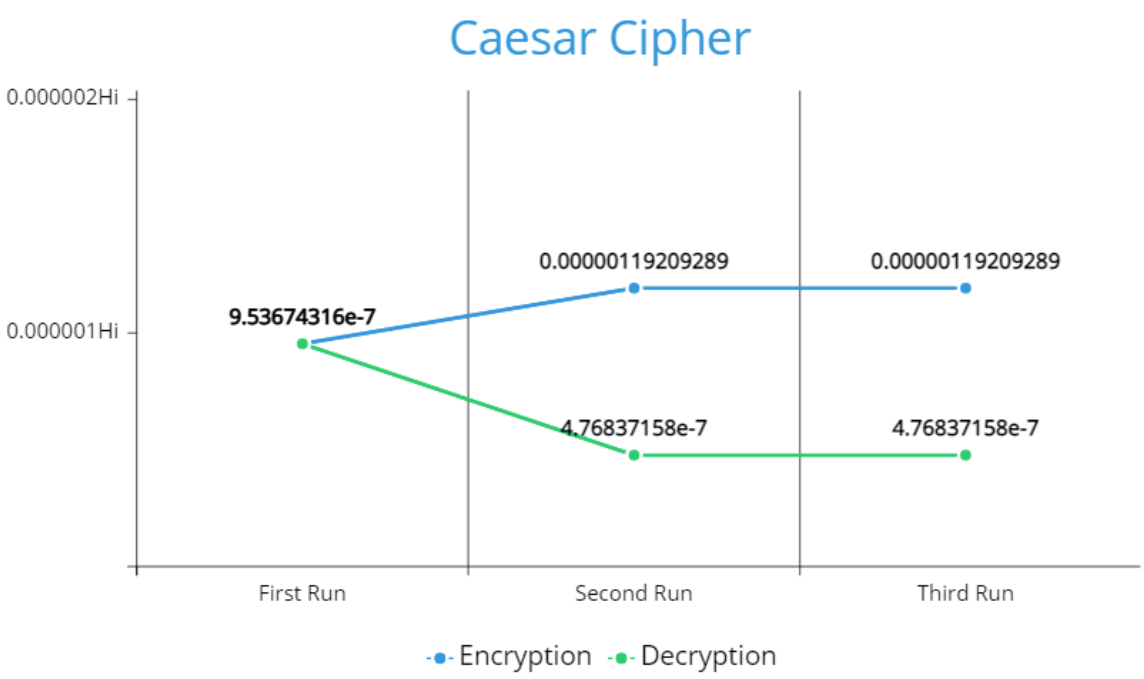
As we can see with the results below, the first run and the last run had equal runtimes while the second run did not have an equal runtime. Like mentioned before, the runtimes may vary. In this case the runtime for run two has a difference of (4.76837158e^-7). We can assume this as an outlier.



Plaintext: The unique thing about it and why it's been used for a while ever since its debut is because it is simple to implement and use and the speed of it is impressive. However shortly after its debut, few vulnerabilities have been found which has rendered it obsolete and non-secure.

Key = 20

Ciphertext: Nby ohckoy nbcha uvion cn uhx qbs cnUm vyyh omyx zil u qbcfy ypl mchwy cnm xyvon cm vywuomy cn cm mcgjfy ni cgjfygyhn uhx omy uhx nby mjyyx iz cn cm cgjlymmcpy Biqypyl mbilnfs uznyl cnm xyvonZ zyq pofhyluvcfcncym bupy vyyh ziohx qbcwb bum lyhxylyx cn ivmifyny uhx hihAmywoly


Caesar Cipher
0.000002Hi
0.000001Hi
0.00000119209289
0.00000119209289
9.53674316e-7
4.76837158e-7
4.76837158e-7
First Run
Second Run
Third Run
Encryption
Decryption