

Cameron Bryant
Assignment 1
CS458

Source code:

```
# Caesar Cipher Encryption, Decryption, and Brute Force Attack

# Function to encrypt plaintext using a shift key
def encrypt(plaintext, key):
    ciphertext = ""
    for char in plaintext:
        if char.isalpha():
            shift = key % 26
            if char.islower():
                ciphertext += chr((ord(char) - ord('a') + shift) % 26 +
ord('a')) #Lower-case handling
            else:
                ciphertext += chr((ord(char) - ord('A') + shift) % 26 +
ord('A')) #Upper-case handling
        else:
            ciphertext += char # Non-alphabetics stay the same
    return ciphertext

# Function to decrypt ciphertext using a shift key
def decrypt(ciphertext, key):
    return encrypt(ciphertext, -key) # Reverse the shift for decryption

# Function to perform a brute force attack on the ciphertext
def brute_force_attack(ciphertext):
    print("Brute force attack results:")
    for key in range(1, 26): #Try every possible combination of ciphertext
and key
        possible_plaintext = decrypt(ciphertext, key)
        print(f"Key = {key}: {possible_plaintext}")

# Main function
def main():
    while True:
        print("\nChoose an option:")
        print("1. Encryption")
```

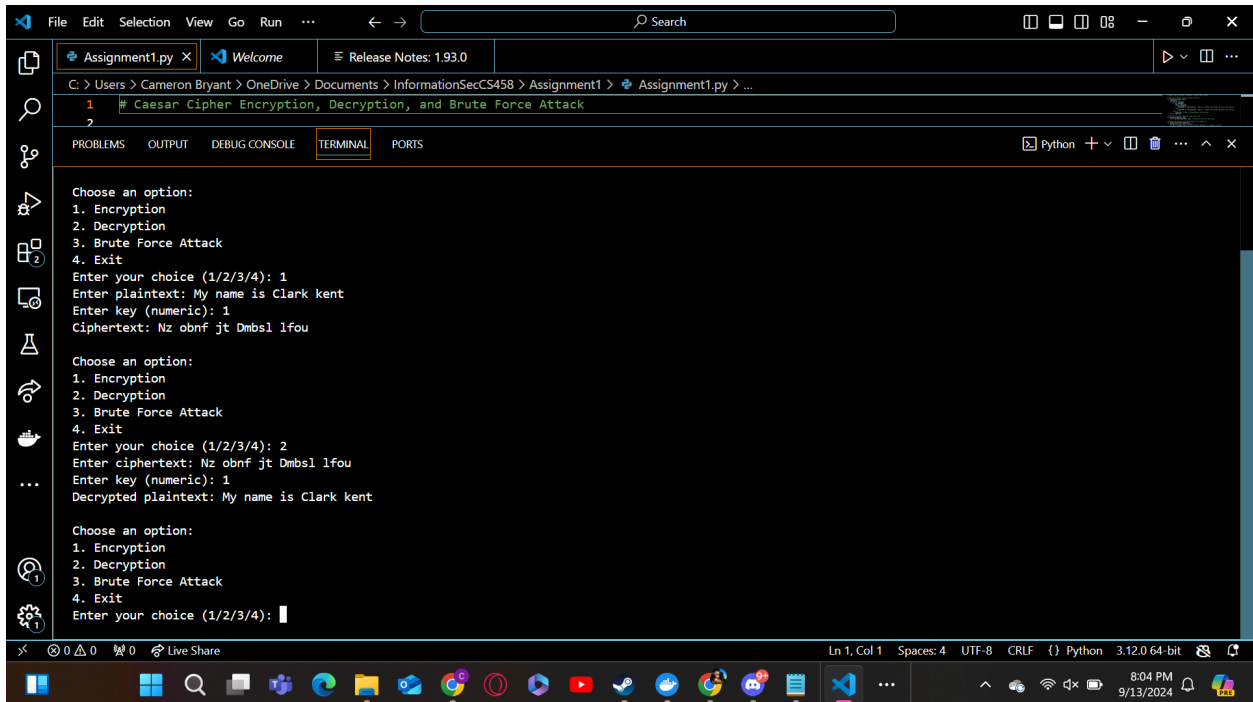
```
print("2. Decryption")
print("3. Brute Force Attack")
print("4. Exit")

choice = input("Enter your choice (1/2/3/4): ") #user inputs
number for option

if choice == '1':
    plaintext = input("Enter plaintext: ")
    key = int(input("Enter key (numeric): "))
    ciphertext = encrypt(plaintext, key)
    print(f"Ciphertext: {ciphertext}")
elif choice == '2':
    ciphertext = input("Enter ciphertext: ")
    key = int(input("Enter key (numeric): "))
    plaintext = decrypt(ciphertext, key)
    print(f"Decrypted plaintext: {plaintext}")
elif choice == '3':
    ciphertext = input("Enter ciphertext: ")
    brute_force_attack(ciphertext)
elif choice == '4':
    break
else:
    print("Invalid choice. Please try again.")

if __name__ == "__main__":
    main()
```

Encryption and Decryption



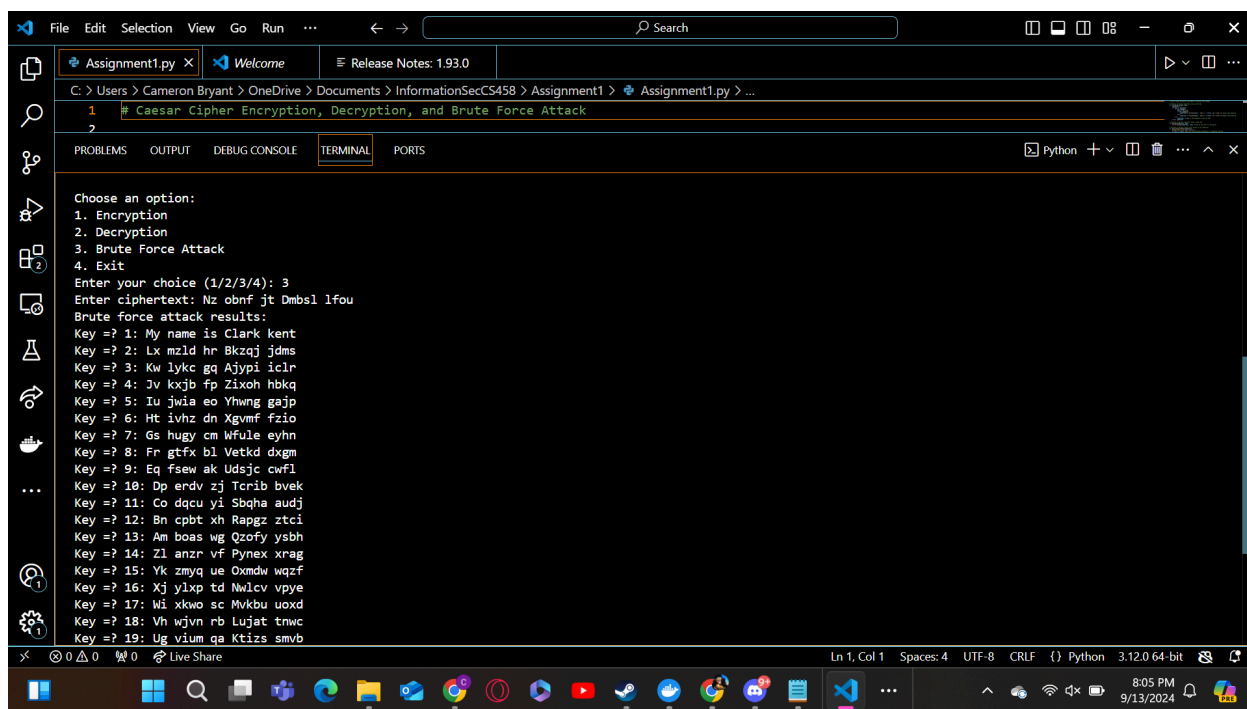
```
1 # Caesar Cipher Encryption, Decryption, and Brute Force Attack
2
Choose an option:
1. Encryption
2. Decryption
3. Brute Force Attack
4. Exit
Enter your choice (1/2/3/4): 1
Enter plaintext: My name is Clark kent
Enter key (numeric): 1
Ciphertext: Nz obnf jt Dmbssl lfou

Choose an option:
1. Encryption
2. Decryption
3. Brute Force Attack
4. Exit
Enter your choice (1/2/3/4): 2
Enter ciphertext: Nz obnf jt Dmbssl lfou
Enter key (numeric): 1
Decrypted plaintext: My name is Clark kent

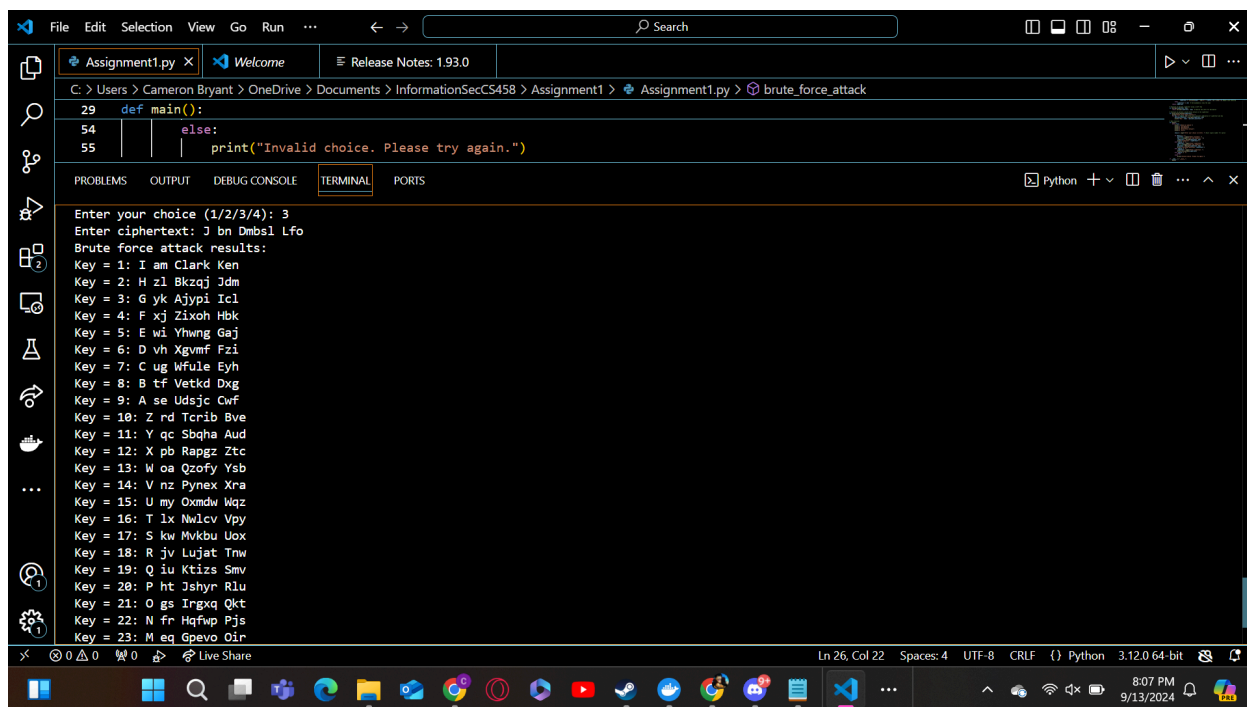
Choose an option:
1. Encryption
2. Decryption
3. Brute Force Attack
4. Exit
Enter your choice (1/2/3/4):
```

The encryption and decryption algorithms are very simple. When encrypting, users are expected to submit a key used for the encryption. After entering the text and key, the function encrypts by cycling the letters in each space as many times as the numeric key. This means when a user uses the letter A and encrypts using key 3, they get the letter D in return. That is because D is 3 char spaces from A in the alphabet. The encryption handles capital and lowercase letters, but not numerical values. Decryption uses the encryption function, but uses the negative of the key. This means it simply performs the shift in the opposite direction of the original encryption. Here, I encrypted and decrypted the phrase “My name is Clark Kent” with the key “1”. It had been encrypted into “Nz obnf jt Dmbssl lfou”. This was returned into its original text using the key used to encrypt it in the first place.

Brute Force Attack



```
1 # Caesar Cipher Encryption, Decryption, and Brute Force Attack
2
Choose an option:
1. Encryption
2. Decryption
3. Brute Force Attack
4. Exit
Enter your choice (1/2/3/4): 3
Enter ciphertext: Nz obnf jt Dmbsl lfou
Brute force attack results:
Key = 1: My name is Clark kent
Key = 2: Lx mzld hr Bkzqj jdms
Key = 3: Kw lykc gq Ajypi iclr
Key = 4: Jv kxjb fp Zixoh hbkq
Key = 5: Iu jwia eo Yhwng gajp
Key = 6: Ht iwhz dn Xgvmf fzio
Key = 7: Gs hugy cm Wfule eyhn
Key = 8: Fr gtfx bl Vetkd dxgm
Key = 9: Eq fsaw ak Udsjc cwf1
Key = 10: Dp erdv zj Tcrib bvek
Key = 11: Co dqcu yi Sdqha audj
Key = 12: Bn cpbt xh Rapgz ztci
Key = 13: Am boas wg Qzofy ysbh
Key = 14: Zl anzr vf Pynex xrag
Key = 15: Yk zmyq ue Oxmdw wqzf
Key = 16: Xj ylxp td Nwlcw vpye
Key = 17: Wi xkwo sc Mvkbu uoxd
Key = 18: Vh wjvn rb Lujat tnwc
Key = 19: Ug vium qa Ktizs smvb
```



```
29 def main():
54     else:
55         print("Invalid choice. Please try again.")
...
Enter your choice (1/2/3/4): 3
Enter ciphertext: J bn Dmbsl Lfo
Brute force attack results:
Key = 1: I am Clark Ken
Key = 2: H z1 Bkzqj Jdm
Key = 3: G yk Ajypi Icl
Key = 4: F xj Zixoh Hbk
Key = 5: E wi Yhwng Gaj
Key = 6: D vh Xgvmf Fzi
Key = 7: C ug Wfule Eyh
Key = 8: B tf Vetkd Dxd
Key = 9: A se Udsjc Cwf
Key = 10: Z rd Tcrib Bve
Key = 11: Y qc Sdqha Aud
Key = 12: X pb Rapgz Ztc
Key = 13: W oa Qzofy Ysb
Key = 14: V nz Pynex Xra
Key = 15: U my Oxmdw Wqz
Key = 16: T lx Nwlcw Vpy
Key = 17: S kw Mvkbu Uox
Key = 18: R jv Lujat Tnw
Key = 19: Q iu Ktizs Smv
Key = 20: P ht Jshyr Rlu
Key = 21: O gs Irgxq Qkt
Key = 22: N fr Hqfwp Pjs
Key = 23: M eq Gpevo Oir
```

The brute force attack works by using the decrypt function but using every possible key there is. It states the used key and the output based on the submitted ciphertext. Here I encrypted the phrases “My name is Clark Kent” and “I am Clark Ken”(typo). Using brute force on both, it attempts every possible key, but at key 1 it shows the correct option, which is the key used to encrypt both.