

# Debugging the Caesar Cipher Program

...

130-[PF]-Lab - Debugging Hello World and Caesar Cipher

# Exercise 1: Working with the buggy Caesar cipher program - Part 1

```
17-01.py
24 def encryptMessage(message, cipherKey, alphabet):
25     encryptedMessage = ""
26     uppercaseMessage = ""
27     for currentCharacter in uppercaseMessage:
28         position = alphabet.find(currentCharacter)
29         newPosition = position + cipherKey
30         if currentCharacter in alphabet:
31             encryptedMessage = encryptedMessage + alphabet[newPosition]
32         else:
33             encryptedMessage = encryptedMessage + currentCharacter
34     return encryptedMessage
35
36 # Decrypt message
37 def decryptMessage(message, cipherKey, alphabet):
38     decryptKey = 1 - int(cipherKey)
39     return encryptMessage(message, decryptKey, alphabet)
40
41 # Main program logic
42 def runCaesarCipherProgram():
43     myAlphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
44     print(f'Alphabet: {myAlphabet}')
45     myAlphabet2 = getDoubleAlphabet(28)
46     print(f'Alphabet2: {myAlphabet2}')
```

```
bash - "lp-10-16-10-72.us"
voclabs:/environment $
```

```
17-01.py - Running
[IKP3db-g] 21:49:14,612332 - INFO - Connected with 127.0.0.1:47706
Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ
Alphabet2: ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZ
Please enter a message to encrypt: hi aws
hi aws
Please enter a key (whole number from 1-25): 3
3
Traceback (most recent call last):
  File "/usr/local/lib64/python3.7/site-packages/ikp3db.py", line 2847, in main
    ikpdb._runscript(mainpyfile)
  File "/usr/local/lib64/python3.7/site-packages/ikp3db.py", line 1526, in _runscript
    exec(statement, globals, locals)
  File "<string>", line 1, in <module>
  File "/home/ec2-user/environment/17 - 01.py", line 56, in <module>
    runCaesarCipherProgram()
  File "/home/ec2-user/environment/17 - 01.py", line 50, in runCaesarCipherProgram
    myEncryptedMessage = encryptMessage(myMessage, myCipherKey, myAlphabet2)
  File "/home/ec2-user/environment/17 - 01.py", line 28, in encryptMessage
    newPosition = position + cipherKey
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

## Problem

- Ran code in debugger
- Received the following error message:
  - “Unsupported operand types(s) for +: ‘int’ and ‘str’”
- Meaning that we cannot add a string with an integer

# Exercise 1: Working with the buggy Caesar cipher program - Part 1

```
10 # Get a message to encrypt
11 def getMessage():
12     stringToEncrypt = input("Please enter a message to encrypt")
13     return stringToEncrypt
14
15 # Get a cipher key
16 def getCipherKey():
17     shiftAmount = input("Please enter a key (whole number from 1-25): ")
18     return shiftAmount
19
20
21 # Encrypt message
22 def encryptMessage(message, cipherKey, alphabet):
23     encryptedMessage = ""
24     uppercaseMessage = message.upper()
25     for currentCharacter in uppercaseMessage:
26         position = alphabet.find(currentCharacter)
27         newPosition = position + int(cipherKey)
28         if currentCharacter in alphabet:
29             encryptedMessage = encryptedMessage + alphabet[newPosition]
30         else:
31             encryptedMessage = encryptedMessage + currentCharacter
32     return encryptedMessage
33
34 # Decrypt message
35 def decryptMessage(message, cipherKey, alphabet):
36     encryptedMessage = message
37     return encryptMessage(encryptedMessage, -cipherKey, alphabet)
```

17-01.py - Stopped

Run Command: 17-01.py Runner: Python 3 CWD: ENV

Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ  
Please enter a message to encrypt: AWS rocks!  
AWS rocks!  
Please enter a key (whole number from 1-25): 3  
Encrypted Message: DZV URFNM!  
Decrypted Message: AWS ROCKS!

Process exited with code: 0

## Solution

- The getCypherKey returns a string
- Therefore turning the cypherKey into an integer within the EncryptMessage function allows it to interact with position of the alphabet

# Exercise 2: Working with the buggy Caesar cipher program - Part 2

```
16 # Get a cipher key
17 def getCipherKey():
18     shiftAmount = input("Please enter a key (whole number from
19     return shiftAmount
20
21 # Encrypt message
22 def encryptMessage(message, cipherKey, alphabet):
23     encryptedMessage = ""
24     uppercaseMessage = ""
25     uppercaseMessage = message
26     for currentCharacter in uppercaseMessage:
27         position = alphabet.find(currentCharacter)
28         newPosition = position + int(cipherKey)
29         if currentCharacter in alphabet:
30             encryptedMessage = encryptedMessage + alphabet[new
31         else:
32             encryptedMessage = encryptedMessage + currentChara
33     return encryptedMessage
34
35 # Decrypt message
36 def decryptMessage(message, cipherKey, alphabet):
37     decryptKey = -1 * int(cipherKey)
38     return encryptMessage(message, decryptKey, alphabet)
39
40 # Main program logic
41 def runCaesarCipherProgram():
42     myAlphabet="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
43
44 # 17-02.py - Running
45 Command: 17-02.py
46 Runner: Python 3 CWD ENV
47
48 [IKP3db-g] 22:01:21,970464 - INFO - IKP3db 1.4.1 - Inux Python Debugger for CPython 3.6+
49 [IKP3db-g] 22:01:21,970781 - INFO - IKP3db listening on 127.0.0.1:15471
50 [IKP3db-g] 22:01:22,007287 - INFO - Connected with 127.0.0.1:143678
51 Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ
52 Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLmnopqrstuvwxyz
53 Please enter a message to encrypt: Hi Neshav
54 Hi Neshav
55 Please enter a key (whole number from 1-25): 3
56 Encrypted Message: K! Neshav
```

## Problem

- The code encrypted the uppercase letters however did not encrypt the lowercase letters

# Exercise 2: Working with the buggy Caesar cipher program - Part 2

```
14 return stringToEncrypt
15
16 # Get a cipher key
17 def getCipherKey():
18     shiftAmount = input("Please enter a key (whole number from
19     return shiftAmount
20
21 # Encrypt message
22 def encryptMessage(message, cipherKey, alphabet):
23     encryptedMessage = ""
24     uppercaseMessage = message.upper()
25     for currentCharacter in uppercaseMessage:
26         position = alphabet.find(currentCharacter)
27         newPosition = position + int(cipherKey)
28         if currentCharacter in alphabet:
29             encryptedMessage = encryptedMessage + alphabet[new
30         else:
31             encryptedMessage = encryptedMessage + currentChara
32     return encryptedMessage
33
34 # Decrypt message
35 def decryptMessage(message, cipherKey, alphabet):
36     decryptKey = -1 * int(cipherKey)
37     return encryptMessage(message, decryptKey, alphabet)
38
39 # Main program logic
40 def runCaesarCipherProgram():
41     runCaesarCipherProgram()
42
43 if __name__ == '__main__':
44     runCaesarCipherProgram()
```

```
17:10:22.py - Running
[IPython] 22:28:48.65862 - INFO - IPython 1.4.1 - Inout Python Debugger for CPython 3.6+
[IPython] 22:28:48.656242 - INFO - IPython listening on 127.0.0.1:15471
[IPython] 22:28:48.635965 - INFO - Connected with 127.0.0.1:144848
Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ
Alphabet2: ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZ
Please enter a message to encrypt: HI Keshav
HI Keshav
Please enter a key (whole number from 1-25): 2
2
Encrypted Message: JK MGJDCX
```

## Solution

- Ensure that all “message” input are converted to uppercase by using “upper()”

# Exercise 3: Working with the buggy Caesar cipher program - Part 3

## Problem

- The code was able to encrypt the message, however failed to decrypt the message accurately

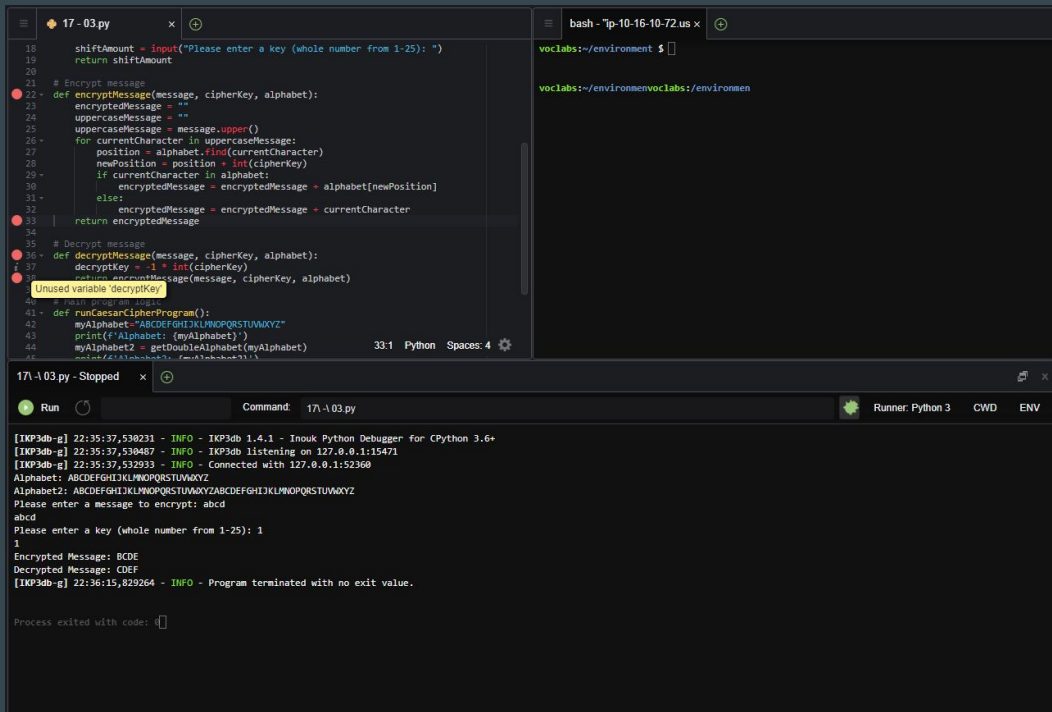
```
17 - 03.py x
30     encryptedMessage = encryptedMessage + alphabet[newPosition]
31     else:
32         encryptedMessage = encryptedMessage + currentCharacter
33     return encryptedMessage
34
35 # Decrypt Message
36 def decryptMessage(message, cipherKey, alphabet):
37     decryptKey = -1 * int(cipherKey)
38     return encryptMessage(message, cipherKey, alphabet)
39
40 # Main program logic
41 def runCaesarCipherProgram():
42     myAlphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
43     print(f'Alphabet: {myAlphabet}')
44     myAlphabet2 = getDoubleAlphabet(myAlphabet)
45     print(f'Alphabet2: {myAlphabet2}')
46     myMessage = getMessage()
47     print(myMessage)
48     myCipherKey = getCipherKey()
49     print(myCipherKey)
50     myEncryptedMessage = encryptMessage(myMessage, myCipherKey, myAlphabet2)
51     print(f'Encrypted Message: {myEncryptedMessage}')
52     myDecryptedMessage = decryptMessage(myEncryptedMessage, myCipherKey, myAlphabet2)
53     print(f'Decrypted Message: {myDecryptedMessage}')
54
55 # Main logic
56 runCaesarCipherProgram()
```

```
bash - "ip-10-16-10-72.us x
voclabs:~/environment $
voclabs:~/environmentvoclabs:/environmen
```

```
17 - 03.py - Stopped x
Run Command: 17 - 03.py Runner: Python 3 CWD ENV
Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ
Alphabet2: ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLmnopqrstuvwxyz
Please enter a message to encrypt: hi keshav
hi keshav
Please enter a key (whole number from 1-25): 2
2
Encrypted Message: JK HGUJCX
Decrypted Message: LM OIMLEZ

Process exited with code: 0
```

# Exercise 3: Working with the buggy Caesar cipher program - Part 3



```
17-03.py x
18 shiftAmount = input("Please enter a key (whole number from 1-25): ")
19 return shiftAmount
20
21 # Encrypt message
22 def encryptMessage(message, cipherKey, alphabet):
23     encryptedMessage = ""
24     uppercaseMessage = ""
25     uppercaseMessage = message.upper()
26     for currentCharacter in uppercaseMessage:
27         position = alphabet.find(currentCharacter)
28         newPosition = position + int(cipherKey)
29         if currentCharacter in alphabet:
30             encryptedMessage = encryptedMessage + alphabet[newPosition]
31         else:
32             encryptedMessage = encryptedMessage + currentCharacter
33     return encryptedMessage
34
35 # Decrypt message
36 def decryptMessage(message, cipherKey, alphabet):
37     decryptKey = 1 - int(cipherKey)
38     return encryptMessage(message, cipherKey, alphabet)
39
40 # Main program logic
41 def runCaesarCipherProgram():
42     myAlphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
43     print(f"Alphabet: {myAlphabet}")
44     myAlphabet2 = getDoubleAlphabet(myAlphabet)
45     # ... (rest of the code) ...
46
47 33:1 Python Spaces: 4
48
49 17-03.py - Stopped x
50
51 Run Command: 17-03.py Runner: Python 3 CWD ENV
52
53 [IKP3db-g] 22:35:37,530231 - INFO - IKP3db 1.4.1 - Inout Python Debugger for CPython 3.6+
54 [IKP3db-g] 22:35:37,530487 - INFO - IKP3db listening on 127.0.0.1:15471
55 [IKP3db-g] 22:35:37,532933 - INFO - Connected with 127.0.0.1:52368
56 [IKP3db-g] 22:35:37,532933 - INFO - Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ
57 Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZ
58 Please enter a message to encrypt: abcd
59 abcd
60 Please enter a key (whole number from 1-25): 1
61 1
62 Encrypted Message: ECDE
63 Decrypted Message: CDEF
64 [IKP3db-g] 22:36:15,829264 - INFO - Program terminated with no exit value.
65
66 Process exited with code: 0
```

## Solution

- First, I checked the encryption to see if there were any discrepancies
- Then proceeded to run the decryptMessage function and there was a prompt: Unused variable 'decryptKey'
- Replaced the 'cipherKey' variable with the 'decryptKey' variable

# Exercise 4: Working with the buggy Caesar cipher program - Part 4

## Problem

- The code was able to encrypt the message but the decrypted message was simply a duplicate of the encrypted message

```
17 - 04.py
30     encryptedMessage = encryptedMessage + alphabet[newPosition]
31 else:
32     encryptedMessage = encryptedMessage + currentCharacter
33 return encryptedMessage
34
35 # Decrypt message
36 def decryptMessage(message, cipherKey, alphabet):
37     decryptKey = -1 * int(cipherKey)
38     return encryptMessage(message, decryptKey, alphabet)
39
40 # Main program logic
41 def runCaesarCipherProgram():
42     myAlphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
43     print(f'Alphabet: {myAlphabet}')
44     myAlphabet2 = getDoubleAlphabet(myAlphabet)
45     print(f'Alphabet2: {myAlphabet2}')
46     myMessage = getMessage()
47     print(myMessage)
48     myCipherKey = getCipherKey()
49     print(myCipherKey)
50     myEncryptedMessage = encryptMessage(myMessage, myCipherKey, myAlphabet2)
51     print(f'Encrypted Message: {myEncryptedMessage}')
52     myDecryptedMessage = decryptMessage(myEncryptedMessage, myCipherKey, myAlphabet2)
53     print(f'Decrypted Message: {myDecryptedMessage}')
54
55 # Main logic
56 runCaesarCipherProgram()
```

56.25 Python Spaces: 4

17 - 04.py - Stopped x

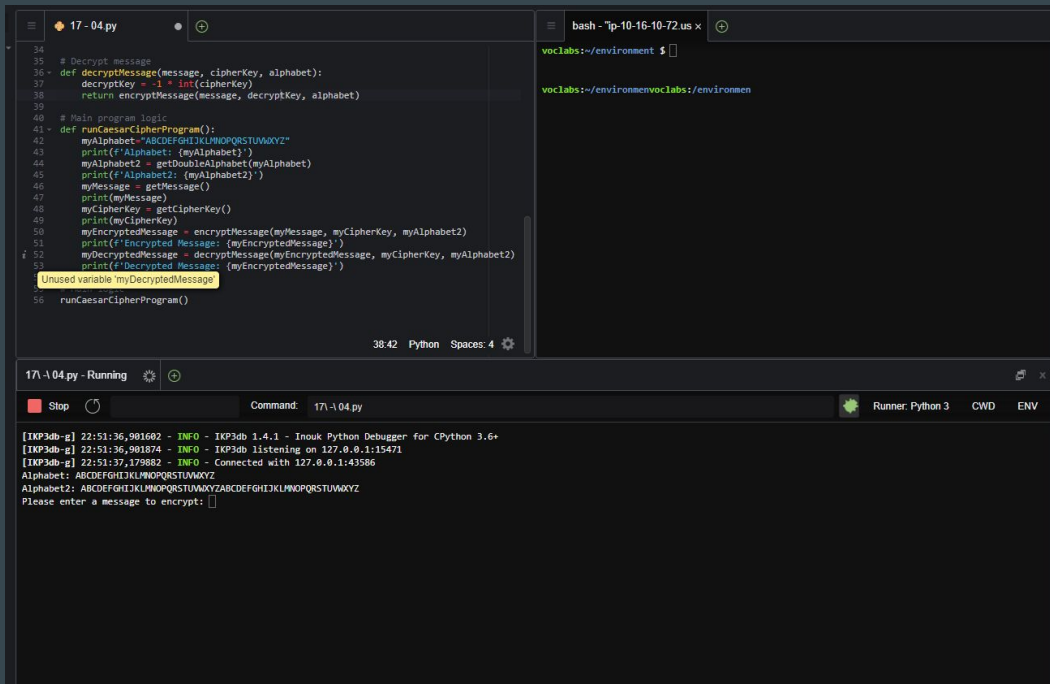
Run Command: 17 - 04.py Runner: Python 3 CWD ENV

Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ  
Alphabet2: ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZ  
Please enter a message to encrypt: abcde  
abcde  
Please enter a key (whole number from 1-25): 1  
1  
Encrypted Message: BCDEF  
Decrypted Message: BCDEF

Process exited with code: 0



# Exercise 4: Working with the buggy Caesar cipher program - Part 4



The screenshot shows a code editor with a Python file named '17-04.py'. The code defines a Caesar cipher program with functions for encryption and decryption. A bug is highlighted: an unused variable 'myDecryptedMessage' is assigned but never used. The terminal output shows the program running successfully, displaying the alphabet and prompting for a message to encrypt.

```
34 # Decrypt message
35
36 def decryptMessage(message, cipherKey, alphabet):
37     decryptKey = -1 * int(cipherKey)
38     return encryptMessage(message, decryptKey, alphabet)
39
40 # Main program logic
41 def runCaesarCipherProgram():
42     myAlphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
43     print(f'Alphabet: {myAlphabet}')
44     myAlphabet2 = getDoubleAlphabet(myAlphabet)
45     print(f'Alphabet2: {myAlphabet2}')
46     myMessage = getMessage()
47     print(myMessage)
48     myCipherKey = getCipherKey()
49     print(myCipherKey)
50     myEncryptedMessage = encryptMessage(myMessage, myCipherKey, myAlphabet2)
51     print(f'Encrypted Message: {myEncryptedMessage}')
52     myDecryptedMessage = decryptMessage(myEncryptedMessage, myCipherKey, myAlphabet2)
53     print(f'Decrypted Message: {myEncryptedMessage}')
54     Unused variable 'myDecryptedMessage'
55
56 runCaesarCipherProgram()
```

Terminal output:

```
bash - "ip-10-16-10-72.us x"
voclabs:~/environment $
voclabs:~/environment voclabs:/environmen
```

## Solution

- There was a prompt stating: Unused variable 'myDecryptedMessage'
- Replaced the {myEncryptedMessage} variable with {myDecryptedMessage}