

Setelah itu kita mencari file yang ada di computer kita.

```
Terminal
rpl-16@rpl16-ThinkCentre-A70: ~/triawan
rpl-16@rpl16-ThinkCentre-A70:~$ cd triawan
rpl-16@rpl16-ThinkCentre-A70:~/triawan$ python chapter7.py
File "chapter7.py", line 15
    for key in range(len(LETTERS)):
    ^
IndentationError: Unexpected indent
rpl-16@rpl16-ThinkCentre-A70:~/triawan$ python chapter7.py
File "chapter7.py", line 57:
33     else:
34         if symbol in LETTERS:
IndentationError: unindent does not match any outer indentation level
rpl-16@rpl16-ThinkCentre-A70:~/triawan$ python chapter7.py
Key #25: HVWG WG AM GSQFSH ASGGOUS.
rpl-16@rpl16-ThinkCentre-A70:~/triawan$ python chapter7.py
Key #25: VHUDQJ MDP 1
rpl-16@rpl16-ThinkCentre-A70:~/triawan$ python chapter7.py
Key #0: UGTCPI LCO 1
Key #1: TFSBOH KBN 1
Key #2: SERANG JAM 1
Key #3: RDQZMF IZL 1
Key #4: QCPYLE HYK 1
Key #5: PBOKKD GXJ 1
Key #6: OANHJC FWI 1
Key #7: NZMVIB EVH 1
The add number's symbol at the end of translated
50     translated = translated + LETTERS[num]
51
52
53
54     else:
55
56 # just add the symbol without encrypting/decrypting
57
58     translated = translated + symbol
59
60
61
62 # display the current key being tested, along with its decryption
63
64     print('Key #%s: %s' % (key, translated))
65
```

```

Terminal
rpl-16@rpl16-ThinkCentre-A70: ~/traiwan
IndentationError: unindent does not match any outer indentation level
rpl-16@rpl16-ThinkCentre-A70:~/traiwan$ python chapter7.py
Key #25: HVWG WG AM GSQFSH ASGGOUS.
rpl-16@rpl16-ThinkCentre-A70:~/traiwan$ python chapter7.py
Key #25: VHWDQJ MDP
rpl-16@rpl16-ThinkCentre-A70:~/traiwan$ python chapter7.py
Key #0: UGTCPI LCO 1
Key #1: TFSBOH KBN 1
Key #2: SERANG JAM 1
Key #3: RDQZMF IZL 1
Key #4: QCPYLE HYK 1
Key #5: PBOXKD GXJ 1
Key #6: OANWJC FWI 1
Key #7: NZMVB EVH 1
Key #8: MYLUHA DUG 1
Key #9: LKKTGZ CTF 1
Key #10: KWJSFY BSE 1
Key #11: JVIREX ARD 1
Key #12: IUHQDW ZQC 1
Key #13: HTGPCV YPB 1
Key #14: GSFOBU XOA 1
Key #15: FRENAT WNZ 1
Key #16: EQDMZS VMY 1
# add number 5 symbol at the end of translated
50
51
52
53
54
55
56 # just add the symbol without encrypting/decrypting
57
58
59
60
61
62 # display the current key being tested, along with its decryption
63
64
65

```

Setelah itu kita cek hasil pesan yang keluar dari python itu.

```
rpl-16@rpl16-ThinkCentre-A70: ~/triawan$ python chapter7.py
Key #25: VHUDQJ MDP 1
```

Terakhir kita ubah dari pesan yang sebelumnya diganti dengan kata “Serang Jam 1”.

```
Terminal
rpl-16@rpl16-ThinkCentre-A70: ~/triawan$ python chapter7.py
Key #0: UGTCPJ LCO 1
Key #1: TFSBOH KBN 1
Key #2: SERANG JAM 1
Key #3: RDQZMF IZL 1
Key #4: QCPYLE HYK 1
Key #5: PBOXKD GXJ 1
Key #6: OANWJC FWI 1
Key #7: NZWJTB EVH 1
Key #8: MYLUHA DUG 1
Key #9: LXXGTZ CTF 1
Key #10: KWDSEY BSE 1
Key #11: JVIREX ARD 1
Key #12: IUHQDW ZQC 1
Key #13: HTGPCV YPB 1
Key #14: GSFOBU XOA 1
Key #15: FRENAT WNZ 1
Key #16: EQDMZS VMY 1
Key #17: DPCLYR ULX 1
Key #18: COBKXQ TKW 1
Key #19: BNAJWP SJV 1
Key #20: AMZIVO RIU 1
Key #21: ZLYHUN QHT 1
Key #22: YKXGTM PGS 1
49 # add number 5 symbol at the end of translated
50
51         translated = translated + LETTERS[num]
52
53     else:
54
55         # just add the symbol without encrypting/decrypting
56
57         translated = translated + symbol
58
59
60
61 # display the current key being tested, along with its decryption
62
63
64     print('Key #%s: %s' % (key, translated))
65
```

Dan ini bentuk dari codingan sebelum kita buat di terminal pada ubuntu dari pertama pengiriman pesan kode palsu (samara) sampai ke pesan kode sebenarnya.

```
# the string to be encrypted/decrypted
message = 'Serang jam 1'

# the encryption/decryption key
key = 2

# tells the program to encrypt or decrypt
mode = 'encrypt' # set to 'encrypt' or 'decrypt'

# every possible symbol that can be encrypted
LETTERS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

# stores the encrypted/decrypted form of the message
translated = ''

# capitalize the string in message
message = message.upper()
```

```
if symbol in LETTERS:

    # get the encrypted (or decrypted) number for this symbol
    num = LETTERS.find(symbol) # get the number of the symbol

    if mode == 'encrypt':
        num = num + key

    elif mode == 'decrypt':
        num = num - key

    # handle the wrap-around if num is larger than the length of
    # LETTERS or less than 0

    if num >= len(LETTERS):
        num = num - len(LETTERS)

    elif num < 0:
        num = num + len(LETTERS)

    # add encrypted/decrypted number's symbol at the end of translated
    translated = translated + LETTERS[num]
```

```
# handle the wrap-around if num is larger than the length of
# LETTERS or less than 0
if num >= len(LETTERS):
    num = num - len(LETTERS)
elif num < 0:
    num = num + len(LETTERS)

# add encrypted/decrypted number's symbol at the end of translated
translated = translated + LETTERS[num]

else:
    # just add the symbol without encrypting/decrypting
    translated = translated + symbol

# print the encrypted/decrypted string to the screen
print(translated)

# copy the encrypted/decrypted string to the clipboard
```

```
# http://inventwithpython.com/hacking (BSD Licensed)

message = 'UGTCPI LCO 1'
LETTERS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

# loop through every possible key
for key in range(len(LETTERS)):

    # It is important to set translated to the blank string so that the
    # previous iteration's value for translated is cleared.
    translated = ''

    # The rest of the program is the same as the original Caesar program:

    # run the encryption/decryption code on each symbol in the message
    for symbol in message:
        if symbol in LETTERS:
            num = LETTERS.find(symbol) # get the number of the symbol

            num = num + len(LETTERS)

            # add number's symbol at the end of translated
            translated = translated + LETTERS[num]

        else:
            # just add the symbol without encrypting/decrypting
            translated = translated + symbol

    # display the current key being tested, along with its decryption
    print('Key #{}: {}'.format(key, translated))
```
