Simple Encryption

Assignment 1
CSSE1001/7030
Semester 2, 2018

Version 1.0.0
10 marks (5%)

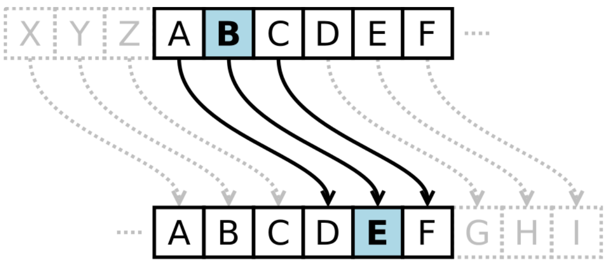
Due Friday 24th August, 2018, 20:30

1. Introduction

This assignment provides you the opportunity to apply foundational concepts taught at the beginning of the course by writing a piece of software to encrypt and decrypt text.

Text is encrypted with a simple cipher where each letter of the alphabet is replaced with another letter that occurs a fixed number of letters later. This fixed number is known as the coding offset.

If the coding offset is 3, for example, `a` is replaced with `d`, `b` is replaced with `e`, `z` is replaced with `c`, etc., as shown below:



Encrypting with a Coding Offset of 3

2. Assignment Tasks

2.1. Example Output

Some example output has been truncated and replaced with ellipses (...). Any lines that have been wrapped onto the next after exceeding the width of this document should be treated as a single line.

```
Welcome to the simple encryption tool!

Please choose an option [e/d/a/q]:
  e) Encrypt some text
  d) Decrypt some text
  a) Automatically decrypt English text
  q) Quit
> e
Please enter some text to encrypt: you will always remember this as the day
Please enter a shift offset (1-25): 7
The encrypted text is: fvb dpss hsdhfz yltltily aopz hz aol khf

Please choose an option [e/d/a/q]:
  e) Encrypt some text
  d) Decrypt some text
  a) Automatically decrypt English text
  q) Quit
> d
Please enter some text to decrypt: asdf ghjkl qwerty uiop z xcvbnm
Please enter a shift offset (1-25): 17
The decrypted text is: jbmo pqstu zfnach drxy i glekwv

Please choose an option [e/d/a/q]:
  e) Encrypt some text
  d) Decrypt some text
  a) Automatically decrypt English text
  q) Quit
> d
Please enter some text to decrypt: a bmkl kso lzw sv sfv lzgmyzl al dggcwv xmf
Please enter a shift offset (1-25): 18
The decrypted text is: i just saw the ad and thought it looked fun

Please choose an option [e/d/a/q]:
  e) Encrypt some text
  d) Decrypt some text
  a) Automatically decrypt English text
  q) Quit
> ep zpv tff uiptf uxp xffwjmt epdups
Invalid command

Please choose an option [e/d/a/q]:
  e) Encrypt some text
  d) Decrypt some text
  a) Automatically decrypt English text
  q) Quit
> a
Please enter some encrypted text: ep zpv tff uiptf uxp xffwjmt epdups
Encryption offset: 1
Decrypted message: do you see those two weevils doctor

Please choose an option [e/d/a/q]:
  e) Encrypt some text
  d) Decrypt some text
  a) Automatically decrypt English text
  q) Quit
> a
Please enter some encrypted text: nmd
Multiple encryption offsets: 4, 12, 21, 25

Please choose an option [e/d/a/q]:
  e) Encrypt some text
  d) Decrypt some text
  a) Automatically decrypt English text
  q) Quit
> a
Please enter some encrypted text: asdf ghjkl qwerty uiop z xcvbnm
No valid encryption offset

Please choose an option [e/d/a/q]:
  e) Encrypt some text
  d) Decrypt some text
  a) Automatically decrypt English text
  q) Quit
> q
Bye!
```

2.2. Getting Started

Download the following files and place them in the same directory:

- `a1.py` : The main assignment file - **write your code here**
- `a1_support.py` : Contains useful supporting functions
- `words.txt` : A list of all words considered English

2.3. Design

You are expected to use good programming practice in your code and you must incorporate at least the following functions. You can assume that:

- text contains only spaces and letters of the English alphabet (both upper & lowercase)
- the coding offset is between 1 and 25, inclusive

2.3.1. encrypt

encrypt(text, offset) -> str

Encrypts `text` by replacing each letter with the letter some fixed number of positions (the `offset`) down the alphabet. Returns the encrypted text.

```
>>> encrypt("you will always remember this as the day", 7)
'fvb dpss hsdhfz yltltily aopz hz aol khf'
>>> encrypt("music is the shorthand of emotion", 2)
'owuke ku vjg ujqtvjcpf qh goqvkqp'
>>> encrypt("qgnrag hgorkey gtj cnovvkj ixkgs", 9)
'zpwajp qpxatnh pcs lwxeets rgtpb'
```

2.3.2. decrypt

decrypt(text, offset) -> str

Decrypts `text` that was encrypted by the `encrypt` function above. Returns the decrypted text.

```
>>> decrypt("a bmkl kso lzw sv sfv lzgmyzl al dggcwv xmf", 17)
'j kvtu tbx uif be boe uipvhiu ju mpplfe gvo'
>>> decrypt("a bmkl kso lzw sv sfv lzgmyzl al dggcwv xmf", 18)
'i just saw the ad and thought it looked fun'
>>> decrypt("asdf ghjkl qwerty uiop z xcvbnm", 17)
'jbmo pqstu zfnach drxy i glekwv'
```

2.3.3. find_encryption_offsets

find_encryption_offsets(encrypted_text) -> tuple<int, ...>

Returns a tuple containing all possible offsets that could have been used if to encrypt some English text into `encrypted_text`.

```
>>> find_encryption_offsets("iynjo fuhsudj ev jxu jycu yj mehai qbb jxu jycu")
(16,)
>>> find_encryption_offsets("vftg amnl aqkkxmn mjmcqlm emkveoxtmn lvbmlomz")
()
>>> find_encryption_offsets("nmd")
(4, 12, 21, 25)
```

2.3.4. main

main() -> None

Handles top-level interaction with user, as outlined in 2.1. Example Output. You may assume that the user will always enter at least one word each time, and that they will always choose a valid offset (between 1 and 25, inclusive).

3. Extension

Extend the program to accept `0` as an input when a shift offset is requested. If the user types this, encrypted/decrypted text should be shown for all offsets (1 to 25, inclusive), as shown below:

```
...
> e
Please enter some text to encrypt: all we can do is smile back
Please enter a shift offset (1-25): 0
The encrypted text is:
  01: bmm xf dbo ep jt tnjmf cbdl
  02: cnn yg ecp fq ku uokng dcem
  03: doo zh fdq gr lv vploh edfn
  04: epp ai ger hs mw wqmpi fego
  05: fqq bj hfs it nx xrnqj gfhp
  06: grr ck igt ju oy ysork hgiq
  07: hss dl jhu kv pz ztpsl ihjr
  08: itt em kiv lw qa auqtm jiks
  09: juu fn ljw mx rb bvrun kjlt
  10: kvv go mkx ny sc cwsvo lkmu
  11: lww hp nly oz td dxtwp mlnv
  12: mxx iq omz pa ue eyuxq nmow
  13: nyy jr pna qb vf fzvyr onpx
  14: ozz ks qob rc wg gawzs poqy
  15: paa lt rpc sd xh hbxat qprz
  16: qbb mu sqd te yi icybu rqsa
  17: rcc nv tre uf zj jdzcv srtb
  18: sdd ow usf vg ak keadw tsuc
  19: tee px vtg wh bl lfbex utvd
  20: uff qy wuh xi cm mgcfy vuwe
  21: vgg rz xvi yj dn nhdgz wvxf
  22: whh sa ywj zk eo oieha xwyg
  23: xii tb zxk al fp pjfib yxzh
  24: yjj uc ayl bm gq qkgjc zyai
  25: zkk vd bzm cn hr rlhkd azbj

Please choose an option [e/d/a/q]:
  e) Encrypt some text
...
```

Further, extend the functions in 2.3. Design to accept all characters (not just English letters and spaces). These characters should be ignored when encrypting or decrypting. When automatically decrypting English text:

- Ignore contractions (words containing an apostrophe - i.e. we're, you've, goose's, etc.)
- Treat words joined together by a hyphen or dash as separate words (i.e. sea-song is treated as two words: sea & song)

For example:

```
...
> a
Please enter some encrypted text: X gtbtbqtg wxb addzxcv gdjcs iwt rdktg pcs lwxhiaxcv id wxbhtau ph wt sxs hd, pcs
iwtc qgtpzxcv dji xc iwpi das htp-hdcv iwpi wt hpcv hd duitc puitglpgsh: "Uxuittc btc dc iwt stps bpc'h rwthi; Nd-
wd-wd, pcs p qdiiat du gjb!"
Encryption offset: 15
Decrypted message: I remember him looking round the cover and whistling to himself as he did so, and then breaking
out in that old sea-song that he sang so often afterwards: "Fifteen men on the dead man's chest; Yo-ho-ho, and a
bottle of rum!"
...
```

4. Marking Criteria

In addition to providing a working solution to the assignment problem, you are required to discuss their submission with a tutor. This discussion will take place in the week following the assignment submission deadline, in the practical session in which you are enrolled. **You must attend this session.** It is treated as an assessment item, so absence without reason may result in an overall mark of zero for this assignment.

For each of the following criteria, part marks will be awarded if the criterion is not fully achieved.

Criteria	Mark
Programming Constructs[1,2]	
Program is well structured and readable	1
Variable and function names are meaningful	1
Algorithmic logic is appropriate	1
Documentation[1,2]	
Entire program is documented clearly and concisely, without excessive or extraneous comments	2
Total	**/10**
Functionality	
Implements correct functionality with no serious errors	4
Extension	
Implements extended functionality correctly	1

> [1]In order to be eligible for the marks for Programming Constructs & Documentation, you must have made a reasonable attempt at implementing at the required functions.
>
> [2]See the course notes on Commenting.

5. Assignment Submission

Your assignment must be submitted via the assignment one submission link on Blackboard. You must only submit the file `a1.py` .

You may submit multiple times prior to the deadline, but only your most recent submission will be marked.

Late submission of the assignment will **not** be accepted. In the event of exceptional circumstances, you may submit a request for an extension.

All requests for extension must be submitted on the UQ Application for Extension of Progressive Assessment form: http://www.uq.edu.au/myadvisor/forms/exams/progressive-assessment-extension.pdf **at least 48 hours prior** to the submission deadline. The application and supporting documentation must be submitted to the ITEE Coursework Studies office (78-425) or by email to enquiries@itee.uq.edu.au.

Change Log

Any changes to this document will be listed here.