

7402 Asn1 User Guide and Testing

"The system is asking for a snake"

Isaac Morneau; A00958405

User Guide	3
Testing	4
Setup	4
Tests	7

User Guide

The application requires a python package for graphing. To install run the following command:

```
`Sudo pip3 install ascii_graph`
```

To use the program itself it's just running the script file with two files as arguments. The first argument is the file to use to find statistical ratios of letters. The second file is the ciphered file to be decrypted.

Testing

Setup

To create ciphered messages the core util tr can be used.

```
17:39:05masterisaac@HMS-Brixford:crap  
↳ tr 'a-z' '[m-zA-1]' < war_and_peace > ciphered_war_and_peace
```

The above will perform a rot13 caesar cipher.

```
17:46:46masterisaac@HMS-Brixford:crap
↳ tail -25 war_and_peace
please visit http://pglaf.org/donate

section    general information about project gutenbergtm electronic
works

professor michael s hart is the originator of the project gutenbergtm
concept of a library of electronic works that could be freely shared
with anyone  for thirty years he produced and distributed project
gutenbergtm ebooks with only a loose network of volunteer support

project gutenbergtm ebooks are often created from several printed
editions all of which are confirmed as public domain in the us unless
a copyright notice is included  thus we do not necessarily keep ebooks
in compliance with any particular paper edition

most people start at our web site which has the main pg search facility
http://www.gutenberg.org

this web site includes information about project gutenbergtm including
how to make donations to the project gutenberg literary archive
foundation how to help produce our new ebooks and how to subscribe to
our email newsletter to hear about new ebooks
17:46:47masterisaac@HMS-Brixford:crap
↳ tail -25 moby_dick
though seated before your evening fire with a poker and
not a harpoon by your side

end of vol i

please do not remove
cards or slips from this pocket

university of toronto library

os melville herman

mobydick

m
```

Above shows the regular text of war and peace as well as moby dick.
Below shows the same text after the ciphering.

```
17:46:50masterisaac@HMS-Brixford:crap
> tail -25 ciphered_war_and_peace
bxqmeq hueuf tffbbsxmrads pazmfq

eqofuaz sqzqdmx uzradymfuaz mnagf bdavqof sgfqznqdsfy qxqofdazuo
iadwe

bdarqeead yuotmqx e tmdf ue ftq adusuzmfad ar ftq bdavqof sgfqznqdsfy
pazqbf ar m xundmdk ar qxqofdazuo iadwe ftmf oagxp nq rdqxxk etmdq
iuft mzkazq rad ftudfk kqmdt tq bdapgoq mzp puefdungfap bdavqof
sgfqznqdsfy qnaawe iuft azxk m xaaeq zqfiadw ar haxgzfqd egbbadf

bdavqof sgfqznqdsfy qnaawe mdq arfz odqmfap rday eqhqdmx bdzfq
qpufuaze mxx ar ituot mdq oazrudyp me bgnxuo paymuz uz ftq ge gzxqee
m oabkdustf zafuoq ue uzoxgpq ftge iq pa zaf zqoqeemduxk wqzb qnaawe
uz oaybxumzoq iuft mzk bmdfuogxmd bmbqd qpufuaz

yaef bqabxq efmdf mf agd iqn eufq ituot tme ftq ymuz bs eqmdot rmouxufk
tffbiiisgfnqdsads

ftue iqn eufq uzoxgpqe uzradymfuaz mnagf bdavqof sgfqznqdsfy uzoxgpuzs
tai fa ymwq pazmfuaze fa ftq bdavqof sgfnqds xufqdmk mduhuq
ragzpmfuaz tai fa tqxb bdapgoq agd zqi qnaawe mzp tai fa egneodunq fa
agd qymux zqixqffq fa tqmd mnagf zqi qnaawe
17:48:05masterisaac@HMS-Brixford:crap
> tail -25 ciphered_moby_dick
ftagst eqmfap nqrdaq kagd qhquzuz rudq iuft m bawqd mzp
zaf m tmdbaaz nk kagd eupq

qzp ar hax u

bxqmeq pa zaf dgyahq
bmdpe ad exube rday ftue baowqf

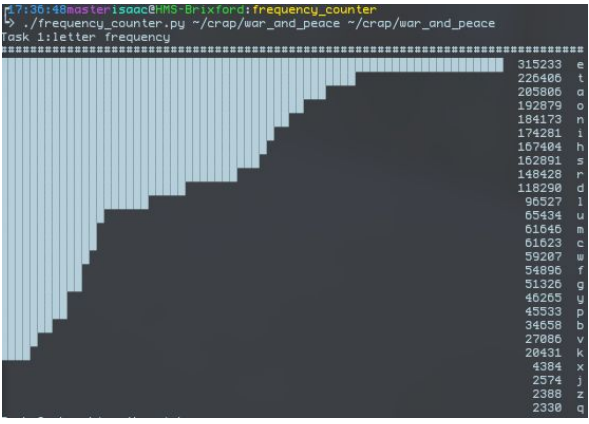
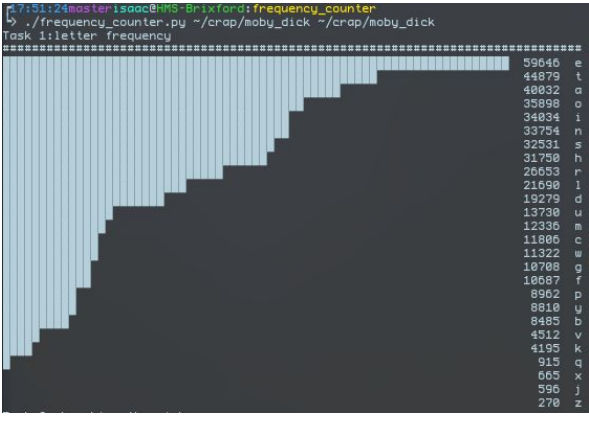
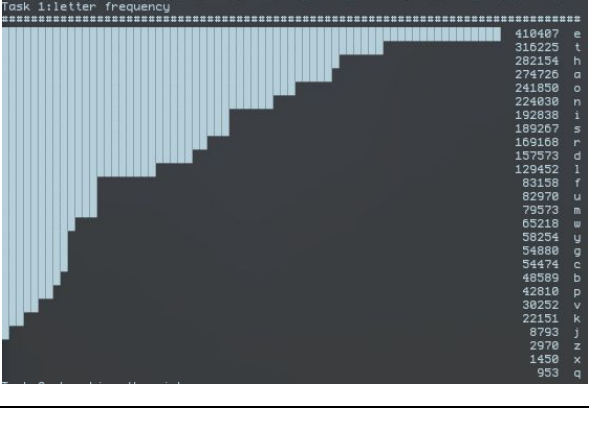
gzuhqdeufk ar fadazfa xundmdk

be yqxhuxxq tqdymz

yankpuow
y
```

These files will be used for the testing. This process was also repeated on the king james bible.

Tests

Test	Steps	Result
Calculate the frequency of war and peace	Pass the file into both arguments	 <pre> 7:36:48master@MS-Brixford:frequency_counter ➤ ./frequency_counter.py ~/crap/war_and_peace ~/crap/war_and_peace Task 1:letter frequency ===== 315233 e 226406 t 205806 a 192879 o 184173 n 174281 i 167484 h 162891 s 148428 r 118290 d 96527 l 65434 u 61646 m 61623 c 59207 w 54896 f 51326 g 46265 y 45533 p 34658 b 27086 v 28431 k 4384 x 2574 j 2388 z 2330 q </pre>
Calculate the frequency of moby dick	Pass the file into both arguments	 <pre> 7:51:24master@MS-Brixford:frequency_counter ➤ ./frequency_counter.py ~/crap/moby_dick ~/crap/moby_dick Task 1:letter frequency ===== 59640 e 44879 t 48832 a 35898 o 34834 i 33754 n 32531 s 31780 h 26653 r 21690 l 19279 d 13730 u 12336 m 11806 c 11322 w 10708 g 10687 f 8962 p 8810 y 8485 b 4512 v 4195 k 915 q 665 x 596 j 270 z </pre>
Calculate the frequency of the king james bible	Pass the file into both arguments	 <pre> Task 1:letter frequency ===== 410407 e 316225 t 282154 h 274726 a 241850 o 224030 n 192838 i 189267 s 169168 r 157573 d 129452 l 83158 f 82970 u 79573 m 65218 w 58254 u 54880 g 54474 c 48589 b 42810 p 38252 v 22151 k 8793 j 2970 z 1450 x 953 q </pre>
Note: While the frequencies are not identical between the files they are very similar across all three		

Check Frequency similarity between war and peace and the king james bible

Pass one into the first argument and pass the other into the second

```
Task 2: breaking the cipher
e is e - 99.72% match
t is t - 99.13% match
a is h - 99.38% match
o is a - 99.10% match
n is o - 99.77% match
i is n - 99.93% match
h is i - 99.37% match
s is s - 99.44% match
r is r - 99.38% match
d is d - 99.78% match
l is l - 99.80% match
u is f - 100.00% match
m is u - 99.86% match
c is m - 99.97% match
w is w - 99.68% match
f is y - 99.64% match
g is g - 99.68% match
y is c - 99.86% match
p is b - 99.71% match
b is p - 99.96% match
v is v - 99.87% match
k is k - 99.88% match
x is j - 99.90% match
j is z - 99.99% match
z is x - 99.95% match
q is q - 99.94% match
```

While the frequencies don't match perfectly they do most of the time or are only off by a couple positions.

Check Frequency similarity between moby dick and the king james bible

Pass one into the first argument and pass the other into the second

```
Task 2: breaking the cipher
e is e - 99.49% match
t is t - 99.39% match
a is h - 99.45% match
o is a - 98.83% match
i is o - 99.47% match
n is n - 99.97% match
s is i - 99.32% match
h is s - 99.37% match
r is r - 99.79% match
l is d - 99.56% match
d is l - 99.93% match
u is f - 99.77% match
m is u - 99.95% match
c is m - 99.95% match
w is w - 99.70% match
g is y - 99.61% match
f is g - 99.51% match
p is c - 99.85% match
y is b - 99.70% match
b is p - 99.59% match
v is v - 99.99% match
k is k - 99.83% match
q is j - 99.91% match
x is z - 99.96% match
j is x - 99.92% match
z is q - 99.97% match
```

While the frequencies don't match perfectly they do most of the time or are only off by a couple positions.

Breaking the cipher text with the original

Pass the original into the frequency and the ciphered into the second option.

```
Task 2: breaking the cipher
q is e - 100.00% match
f is t - 100.00% match
t is h - 100.00% match
m is a - 100.00% match
a is o - 100.00% match
z is n - 100.00% match
u is i - 100.00% match
e is s - 100.00% match
d is r - 100.00% match
p is d - 100.00% match
x is l - 100.00% match
r is f - 100.00% match
g is u - 100.00% match
y is m - 100.00% match
i is w - 100.00% match
k is y - 100.00% match
s is g - 100.00% match
o is c - 100.00% match
n is b - 100.00% match
b is p - 100.00% match
h is v - 100.00% match
w is k - 100.00% match
v is j - 100.00% match
l is z - 100.00% match
j is x - 100.00% match
c is q - 100.00% match
```

When the frequencies line up it can perfectly reverse the cipher applied to it