Et tu, Brute?

Write a Python program that implements a generic Caesar (shift) cipher that takes a ciphertext as input from `stdin` and outputs the correct plaintext and corresponding alphabet shift used in the cipher to `stdout`.

Since you will be playing the part of cryptanalysts, your program will need to generate candidate plaintexts from possible shifts of a provided alphabet. Note that the alphabet can be hard-coded in your program. Once candidate plaintexts are generated, you must filter out invalid ones by comparing "words" in them with words in a provided dictionary. If enough words match (say, three-fourths of them), a candidate plaintext becomes likely. You can also consider using the average length of words in English US to help identify candidate plaintexts.

There are several ways to generate candidate plaintexts. The naive way is to generate them all by iterating through all possible shifts of the alphabet. Another is to use character frequencies in English US to help prioritize shifts of the alphabet. Feel free to use the method(s) of your choice.

Notes and Requirements:

- Submit your source code only. I will provide my own ciphertexts to test with;
- Read the ciphertext from `stdin`;
- Write the plaintext and shift to `stdout`;
- Comment your source code appropriately;
- If the ciphertext contains characters not in the alphabet (e.g., \n), leave them as is in the candidate plaintexts; and
- The ciphertext could contain multiple lines.

Please, no GUIs. Make this a command line application without frills that I can execute at the command line as illustrated below via several sample runs of my program:

```
jgourd@latech:˜$ python et-tu-brute.py < ciphertext-1.txt
SHIFT=13:
The lady said, "Oh my!  You have nice eyes.  Are they yours?"
I laughed.

jgourd@latech:˜$ python et-tu-brute.py < ciphertext-2.txt
SHIFT=87:
Never attribute to malice that which is adequately explained by stupidity.
(Hanlon’s razor)
```