Bookmarks

- Welcome to the edX Platform
- Entrance Survey
- Download Python and Get Motivated!
- Week 1: Python Basics
- Week 2: Simple Programs
- Week 3: Structured Types
- Week 4: Good Programming Practices
- Midterm Exam
- **Week 5: Object Oriented Programming**

Week 5: Object Oriented Programming > Problem Set 5 > Problem 3 - CiphertextMessage

Problem 3 - CiphertextMessage

Bookmark this page

Problem 3 - CiphertextMessage

15.0 points possible (graded)

For this problem, the graders will use our implementation of the `Message` and `PlaintextMessage` classes, so don't worry if you did not get the previous parts correct.

Given an encrypted message, if you know the shift used to encode the message, decoding it is trivial. If `message` is the encrypted message, and `s` is the shift used to encrypt the message, then `apply_shift(message, 26-s)` gives you the original plaintext message. Do you see why?

The problem, of course, is that you don’t know the shift. But our encryption method only has 26 distinct possible values for the shift! We know English is the main language of these emails, so if we can write a program that tries each shift and maximizes the number of English words in the decoded message, we can decrypt their cipher! A simple indication of whether or not the correct shift has been found is if most of the words obtained after a shift are valid words. Note that this only means that most of the words obtained are actual words. It is possible to have a message that can be decoded by two separate shifts into different sets of words. While there are various strategies for deciding between ambiguous decryptions, for this problem we are only looking for a simple solution.

Fill in the methods in the class `CiphertextMessage` acording to the specifications in ps6.py. The methods you should fill in are:

- `__init__(self, text)`: Use the parent class constructor to make your code more concise.
- `decrypt_message(self)`: You may find the helper function `is_word(wordlist, word)` and the string method `split()` useful. Note that `is_word` will ignore punctuation and other special characters when considering whether a word is valid.

9. Classes and Inheritance
Finger Exercises

10. An Extended Example
Finger Exercises

Problem Set 5
Problem Set due Mar 2, 2017 15:30 PST

▸ Week 6: Algorithmic Complexity

▸ Sandbox

Hints

Using string.split

You may find the function `string.split` useful for dividing the text up into words.

```
>>> 'Hello world!'.split('o')
['Hell', ' w', 'rld!']
>>> '6.00.1x is pretty fun'.split(' ')
['6.00.1x', 'is', 'pretty', 'fun']
```

Paste your implementation of the entire `CipertextMessage` class in the box below.

Press ESC then TAB or click outside of the code editor to exit

Unanswered

Submit You have used 0 of 30 attempts

Problem 3 - CiphertextMessage

Topic: Problem Set 5 / Topic-Level Student-Visible Label

Show Discussion