

Exam2D

[New Attempt](#)

Due Wednesday by 6:01pm **Points** 70 **Submitting** a file upload **File Types** zip
Available Oct 27 at 4:40pm - Oct 28 at 6pm 1 day

Problem statement: There are many encryption / decryption techniques that take a string of text and code / decode them according to some algorithm. Some are shift ciphers that simply change the characters by shifting them by a certain number. You have to create an application that implements two of these shift ciphers:

1. **Caesar:** a simple shift cipher (called the Caesar cipher if shift is 3, but here the shift can be any number). The Caesar cipher shifts each character by a fixed number. For key = 2, for example, encrypting the letter a gives c (In ASCII, a is 97, so $97+2=99=c$). Every letter of the message is shifted by the same amount, wrapping around from z back to a – so y, when shifted by 2, would encrypt to a.
2. **Keyword :** The keyword cipher is also a shift cipher, but it uses a key string to shift each character by a different number, using the key's letters positionally. For example, if the message is pqr and the key is bcd, p is shifted by 2 as b is the 2nd alphabet, q is shifted by 3 as c is the 3rd alphabet, and r is shifted by 4 as d is the 4th alphabet, resulting in encrypted code as rtv. If the message was longer, then the key is repeated. So if the message was hello and key was bcd, then the encrypted code would become: jhpnr. As with the Caesar, the shift wraps from z back to a.

Decryption using each of these algorithms simply reverses the encryption process by undoing the shifts.

Important:

- To keep it simple, assume that
 - Messages given to Messenger to encrypt or decrypt are written in lower case alphabets i.e. a-to-z.
 - Any spaces in a message are encrypted as spaces
 - A Caesar key is always a positive integer including 0.
 - A keyword key is always one or more alphabetical characters in lower case

Choose ...

1. Caesar encryption
2. Keyword encryption
3. Caesar decryption
4. Keyword decryption

1

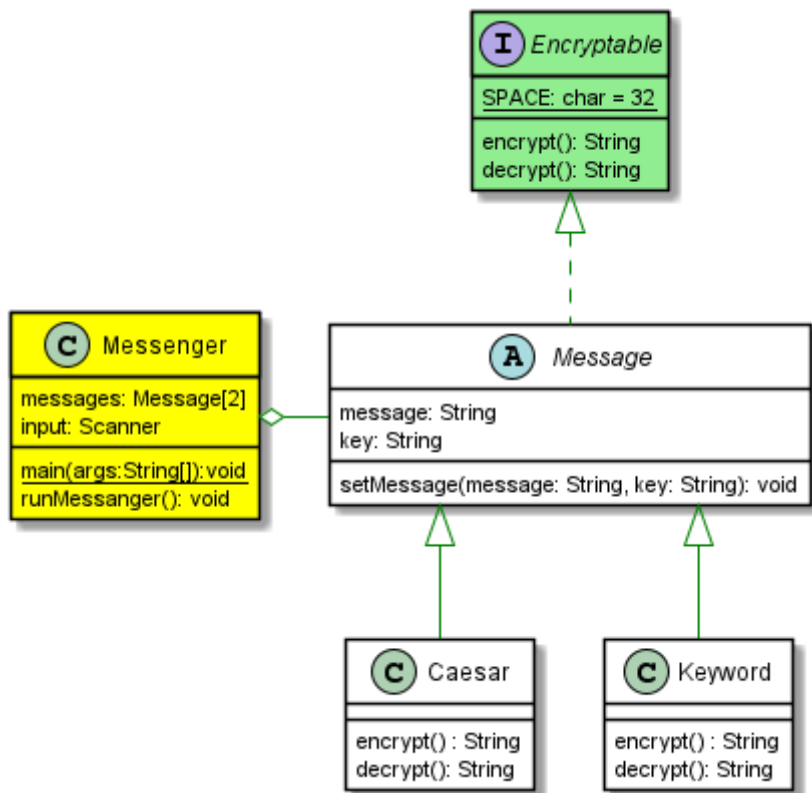
Choose ...

1. Caesar encryption
2. Keyword encryption
3. Caesar decryption
4. Keyword decryption

3

<p>Enter message to encrypt</p> <p>java is great</p> <p>Enter key</p> <p>7</p> <p>Encrypted message: qhch pz nylha</p> <p>Scenario 1: Caesar encryption</p>	<p>Enter message to decrypt</p> <p>qhch pz nylha</p> <p>Enter key</p> <p>7</p> <p>Decrypted message: java is great</p> <p>Scenario 2: Caesar decryption</p>
<p>Choose ...</p> <ol style="list-style-type: none"> 1. Caesar encryption 2. Keyword encryption 3. Caesar decryption 4. Keyword decryption <p>2</p> <p>Enter message to encrypt</p> <p>java is great</p> <p>Enter key</p> <p>abc</p> <p>Encrypted message: kcyb lt jsgdu</p> <p>Scenario 3: Keyword encryption</p>	<p>Choose ...</p> <ol style="list-style-type: none"> 1. Caesar encryption 2. Keyword encryption 3. Caesar decryption 4. Keyword decryption <p>4</p> <p>Enter message to decrypt</p> <p>kcyb lt jsgdu</p> <p>Enter key</p> <p>abc</p> <p>Decrypted message: java is great</p> <p>Scenario 4: Keyword decryption</p>




Solution Design: This design has four classes and one interface as shown in the diagram below. The interface `Encryptable` is fully coded. `Messenger` is partially coded. You need to complete the `Messenger` and code rest of the files as needed. It is important that you follow the design and demonstrate correct use of inheritance, abstraction, and polymorphism. Please refer to comments in the given code files for more details.



Design constraints:

1. Use of Collection classes is not allowed.
2. You must demonstrate the use of polymorphism correctly.

Instructions:

- Download the following files and copy them in exam2 package:
 - [Messenger.java](https://canvas.cmu.edu/courses/25253/files/7050773/download?download_frd=1)  (https://canvas.cmu.edu/courses/25253/files/7050773/download?download_frd=1)
 - [Encryptable.java](https://canvas.cmu.edu/courses/25253/files/7050772/download?download_frd=1)  (https://canvas.cmu.edu/courses/25253/files/7050772/download?download_frd=1)
 - [TestMessages.java](https://canvas.cmu.edu/courses/25253/files/7155644/download?download_frd=1)  (https://canvas.cmu.edu/courses/25253/files/7155644/download?download_frd=1)
- Complete the code as needed.
- Write your name and Andrew-ID in Messenger, Message, Caesar, and Keyword files. Zip these four files into AndrewId-Exam2.zip and submit

Rubric:

- Test case: 36 (12 x 3 points each)
- Console output: 28
- Correct use of polymorphism, abstraction, etc.: 6 points
- Any submission issues may cost up to 2 points.

