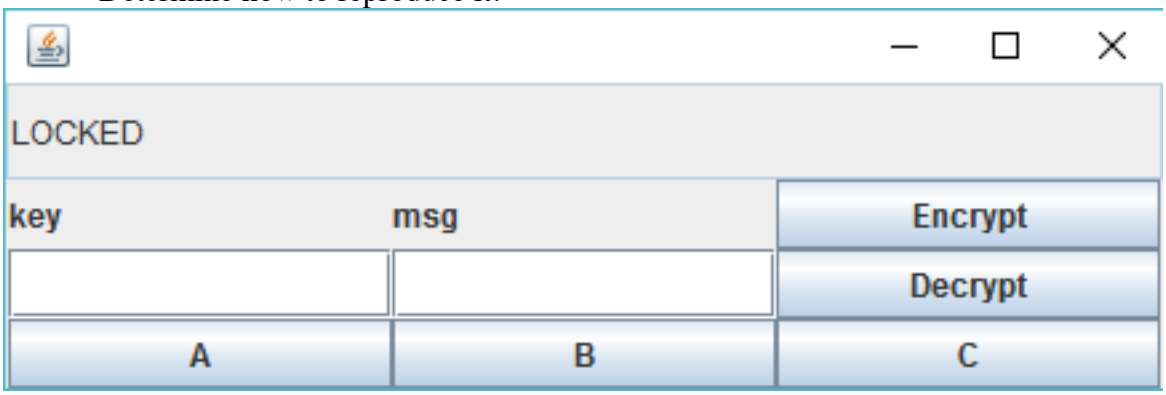
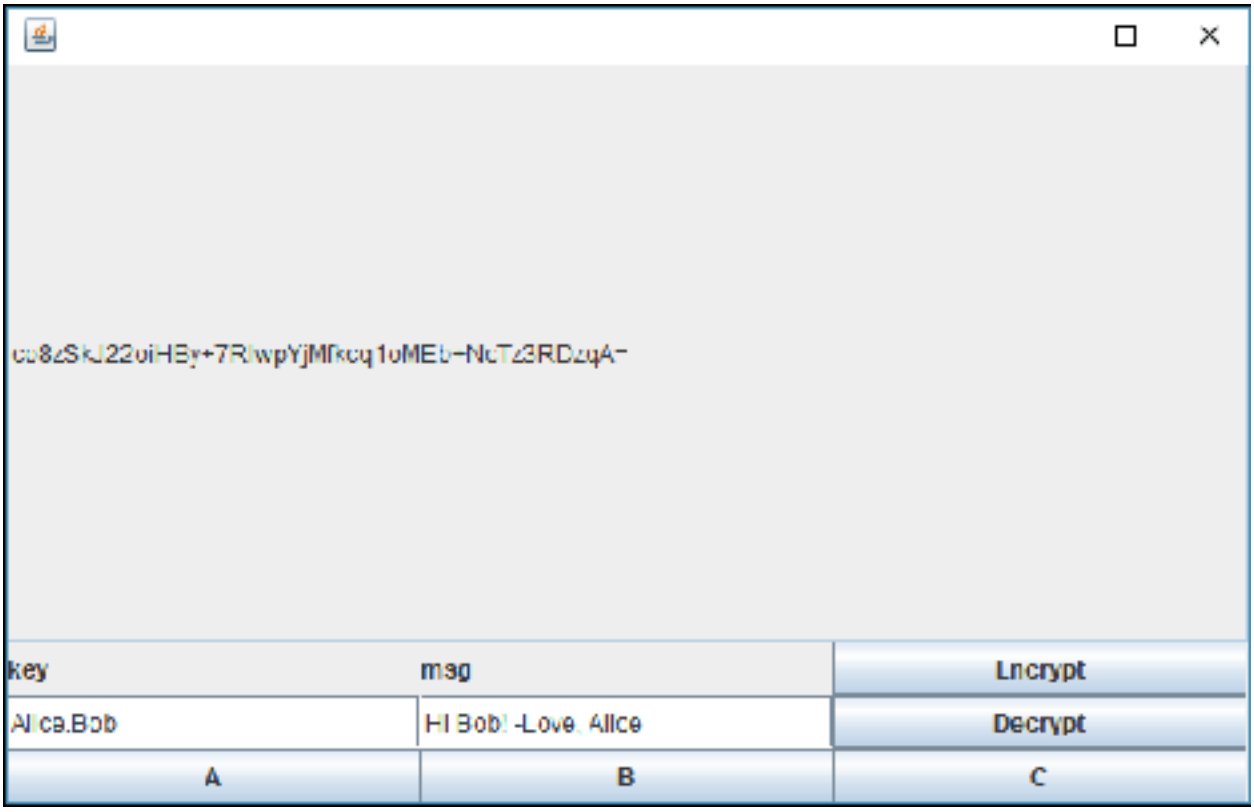
Step 1)

- Examine the UI in the provided screen shots.
- Determine how to reproduce it.





Step 2)

- Make sure that the provide code will compile in your environment.
- Eclipse users: add the provided .jar file to your project as a library.
- Command line users: place the provided .class files in the directory with your .java files

Step 3)

- Use the provided SecretComp to reproduce the interface.
- You will only need to edit SecretComp for this assignment, and you can accomplish everything in the constructor if you use lambda expressions.
- Make the output field at the top a text field (not a label), but make in non-editable. This will allow for copying text from it.

Step 4)

- Attach listeners to each of the 3 buttons labelled A, B, C
- Each of these listeners should pass the button's label to the Secret object's unlock(String) method, then display the message from Secrets' getMessage in the output field

Step 5)

- Attach a listener to the "key" text input field
- If Secrets is locked, it should pass its contents to unlock() on Secrets, then display the message from Secrets' getMessage() in the output field

Step 6)

- Attach listeners to the encrypt & decrypt buttons
- These listeners should call encrypt / decrypt as appropriate on Secrets using the given key and msg in the text fields
- Change the output field to be whatever they return

Step 7)

- Finally, set the output field to be whatever Secrets' getMessage() returns

Step 8)

- Your GUI should now be functional
- Type the word "unlock" into the text field labeled "key"
- A sequence of letters will appear in the label. If not, revisit step 5.
- Press the buttons in order as indicated by the label.
- A "secret key" will then appear in the label. If not, revisit step 4.

Step 9)

- Copy/paste (or type) the displayed secret key into the "key" text field.
- Copy/paste this cipher-text into the "msg" field: **cgmMbvbyDONppeM2paS+7A==**
- Press the decrypt button. The decrypted plain-text should appear in the output field. If not, revisit step 6.
- Fill in the fields under the heading "STEP 9" at the end of your SecretComp.java file.

Step 10)

- Create a different message for a classmate.

- Use YourName.TheirName as the secret key. eg:
    - key: Alice.Bob
    - plain-text: "Hi Bob! -Love, Alice"
    - cipher-text: co8zSkJ22oiHBy+7RIwpYjMfkcq1oMEb+NcTz3RDzqA=
- Note the key, cipher-text, and plain-text of the message that you created at the end of your SecretComp.java file under the heading "STEP 10".
- Send this message to your classmate.

Step 11)

- Record at least one key, cipher-text, and plain-text message set that you received from a classmate at the end of your SecretComp.java file under the heading "STEP 11".

Step 12)

- Using any secret key, encrypt a message for the marker.
- Include the key and the cipher-text at the end of your SecretComp.java under the heading "STEP 12", but do not include the plain-text.