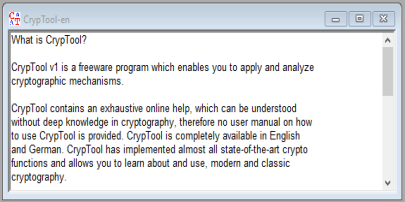
**Network &Information Security (CT-460) \_\_ Lab Session 02**

|  |
| --- |
| **Lab Session 02** |

**OBJECT: Implementation of Caesar Cipher and Rot-13.**

This lab provides an example illustrating the use of the Caesar encryption algorithm. To make it easier to follow the steps that need to be performed with CrypTool, the example is illustrated with a number of screenshots.

First of all, to acquaint ourselves with the Caesar encryption algorithm we will open a document, encrypt it and then decrypt it again. We will then try to get the computer to work out the key with which a plaintext is encrypted. Therefore, we open the file CrypTool-en.txt out of the directory CrypTool\examples via the menu File \ Open.



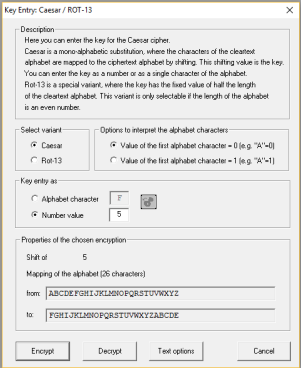
We will now encrypt this document using the Caesar encryption algorithm. To do this, we select the menu Encrypt/Decrypt \ Symmetric (classic) \ Caesar/Rot-13. Then the following which this dialog box appears. The key we enter is the letter F. Additionally, we change the options how to interpret the alphabet characters: The first alphabet character is set to 0.

Department of Software Engineering

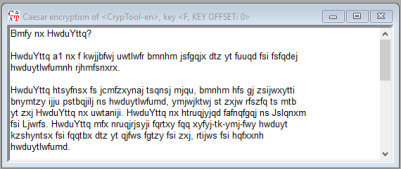
NED University of Engineering & Technology

Page **8** of **75**

**Network &Information Security (CT-460) \_\_ Lab Session 02**

****

Clicking on the **Encrypt** button opens a new window that contains the encrypted text. On closer examination of the text it becomes apparent that the letters have been shifted by 5 position, so that the initial word, CrypTool, is now HwduYttq.



Department of Software Engineering

NED University of Engineering & Technology

Page **9** of **75**

**Network &Information Security (CT-460) \_\_ Lab Session 02**

The plaintext version of this encrypted document can now be obtained by selecting Encrypt/Decrypt \ Classical \ Caesar again. In the dialog box which now appears we enter the key with which the document was encrypted (F). This time we do not want the text to be encrypted, but instead to be decrypted. Therefore the **Decrypt** button must be selected.

Clicking on the **Decrypt** button tells CrypTool to go ahead and decrypt the text. The plaintext appears immediately. We have seen how a text is encrypted using the Caesar encryption algorithm and then decrypted again.

Under the classical Caesar encryption algorithm only the letters are encrypted. During encryption, the lower case letters were converted to upper case letters and encrypted, but all the other characters such as punctuation characters and formatting characters (blank characters and line breaks) were omitted.

In the classical encryption algorithms CrypTool retains the formatting by default. This can be disabled via the menu option Options \ Text Options. In the following dialog box the option Keep characters not present in the alphabet unchanged and the option Distinguish between uppercase and lowercase must be disabled.

We now return to the window containing the plaintext (by clicking on it with the mouse) and encrypt this document once again with the Caesar encryption algorithm, using the key F. From the results one can see immediately that the encrypted text has the same content as the previous version but this time it contains only upper case letters and the formatting has been removed. For example, the first characters in the encrypted text are once again HWDUYTTQ (the ciphertext of CrypTool).

We would like to save this document under a different name. Select File \ Save As to access the dialog box used for saving a document under a new name. As file name we enter CrypTool.Caesar.txt.

The decryption operation works in exactly the same way as described above. We decrypt this document once again and the plaintext version is restored. This of course now consists only of a sequence of upper case letters. Because no distinction is made any longer between upper and lower case letters, and especially because of the absence of formatting, this text is more difficult to read. However, if we compare the document briefly with the original text, we can see that it is the same text. The original text begins with the words, ”CrypTool is a program which will enable you …” These words appear once again in the first line of the decrypted text.

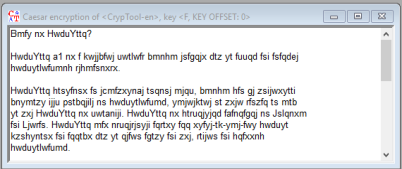
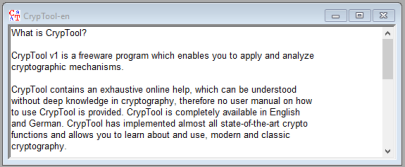
Having now learned the steps involved in encrypting and decrypting a document, we would like to take a look at the security of the Caesar encryption algorithm. To do this, all the windows should be closed apart from the window containing the plaintext and the version of the encrypted text in which the formatting is retained. The only two windows now open should be the following:

Department of Software Engineering

NED University of Engineering & Technology

Page **10** of **75**

**Network &Information Security (CT-460) \_\_ Lab Session 02**

****

First, make the window containing the plaintext the active window again by clicking on it with the mouse. We now have the frequency distribution of the letters (Analysis \ General \ Histogram) calculated. We repeat the same procedure after making the other window active.

When we examine the histogram of the encrypted document we can see that the letter frequencies have merely been shifted by five position. That means that this old encryption algorithm is not secure. Moreover there are only 26 possible keys. The Caesar encryption algorithm can be broken easily by a ciphertext-only attack. To perform such an attack, restore the window containing the encrypted text to the active window and select Analysis \ Ciphertext only \ Caesar. The text will automatically be analyzed.

By analyzing the superposition it is possible to discover the key which was used to encrypt this document. In this case it was the letter F.When you click on the **Decrypt** button in the message window, the plaintext appears, i.e. the text that has been decrypted with the key F that was discovered. So CrypTool managed successfully to find the key with which the document had been encrypted. The only information it needed to do this was the encryption algorithm.

Department of Software Engineering NED University of Engineering & Technology Page **11** of **75**

**Network &Information Security (CT-460) \_\_ Lab Session 02**

**EXERCISE**

**6. Explain the working of Caesar cipher and rot-13.**

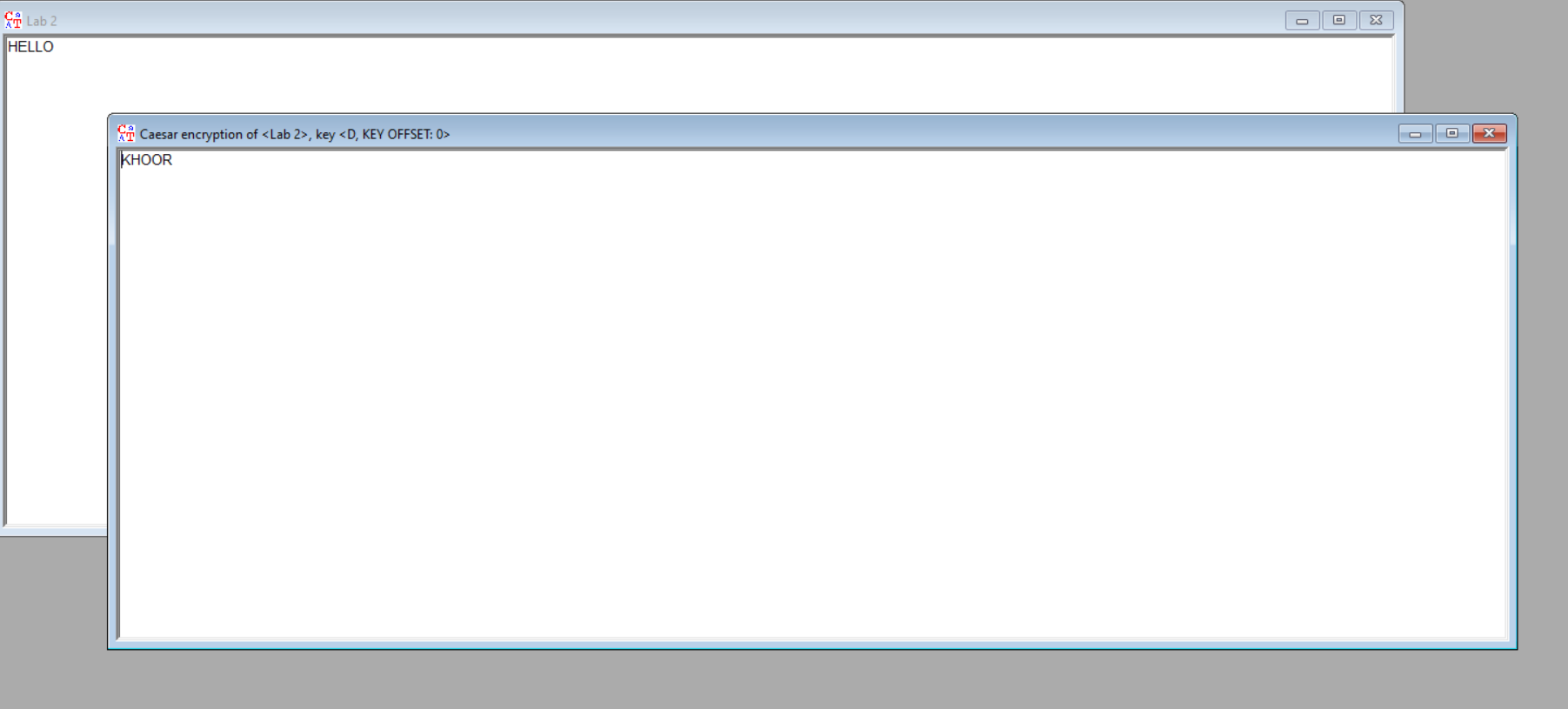
**Caesar Cipher**

The Caesar Cipher is a simple encryption technique where each letter in the plaintext is shifted by a certain number of positions in the alphabet. For example, with a shift of 3:

* **A** becomes **D**
* **B** becomes **E**
* **C** becomes **F**

Example with a shift of 3:

* Plaintext: **HELLO**
* Cipher text: **KHOOR**



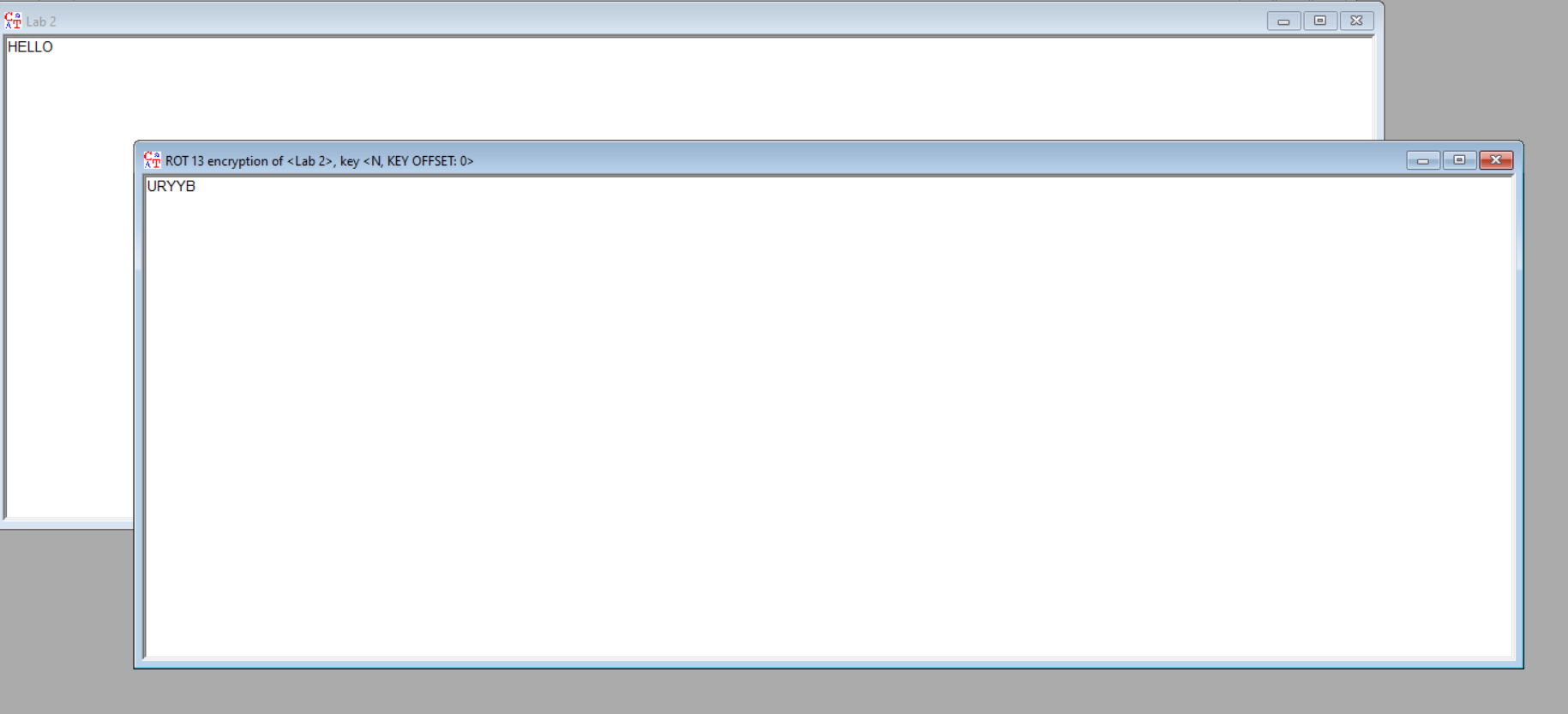
**ROT-13**

ROT-13 is a specific case of the Caesar Cipher where the shift value is fixed at 13. It means every letter is shifted by 13 positions.

* **A** becomes **N**
* **B** becomes **O**
* **C** becomes **P**

Example:

* Plaintext: **HELLO**
* ROT-13 Cipher text: **URYYB**



**7. What is the difference between rot-13 and Caesar. How can Caesar cipher become rot-13?**

|  |  |  |
| --- | --- | --- |
| Feature | Caesar Cipher | ROT-13 |
| Shift Value | Any chosen value (e.g., 1, 3, 7, etc.) | Fixed at 13 |
| Encryption Formula | Shift each letter by the chosen value | Shift each letter by 13 |
| Decryption Process | Reverse the shift (subtract the shift value) | Same process as encryption (apply ROT-13 again) |
| Flexibility | User can choose any shift value | Fixed to 13, no flexibility |
| Symmetry | Encryption and decryption require different shifts | Encryption and decryption are the same |
| Common Use | General encryption with custom shift | Used for simple encoding, often in jokes |
| Alphabet Size Dependence | Works with any alphabet size | Specifically designed for 26-letter alphabets (English) |

**How can Caesar cipher become rot 13**

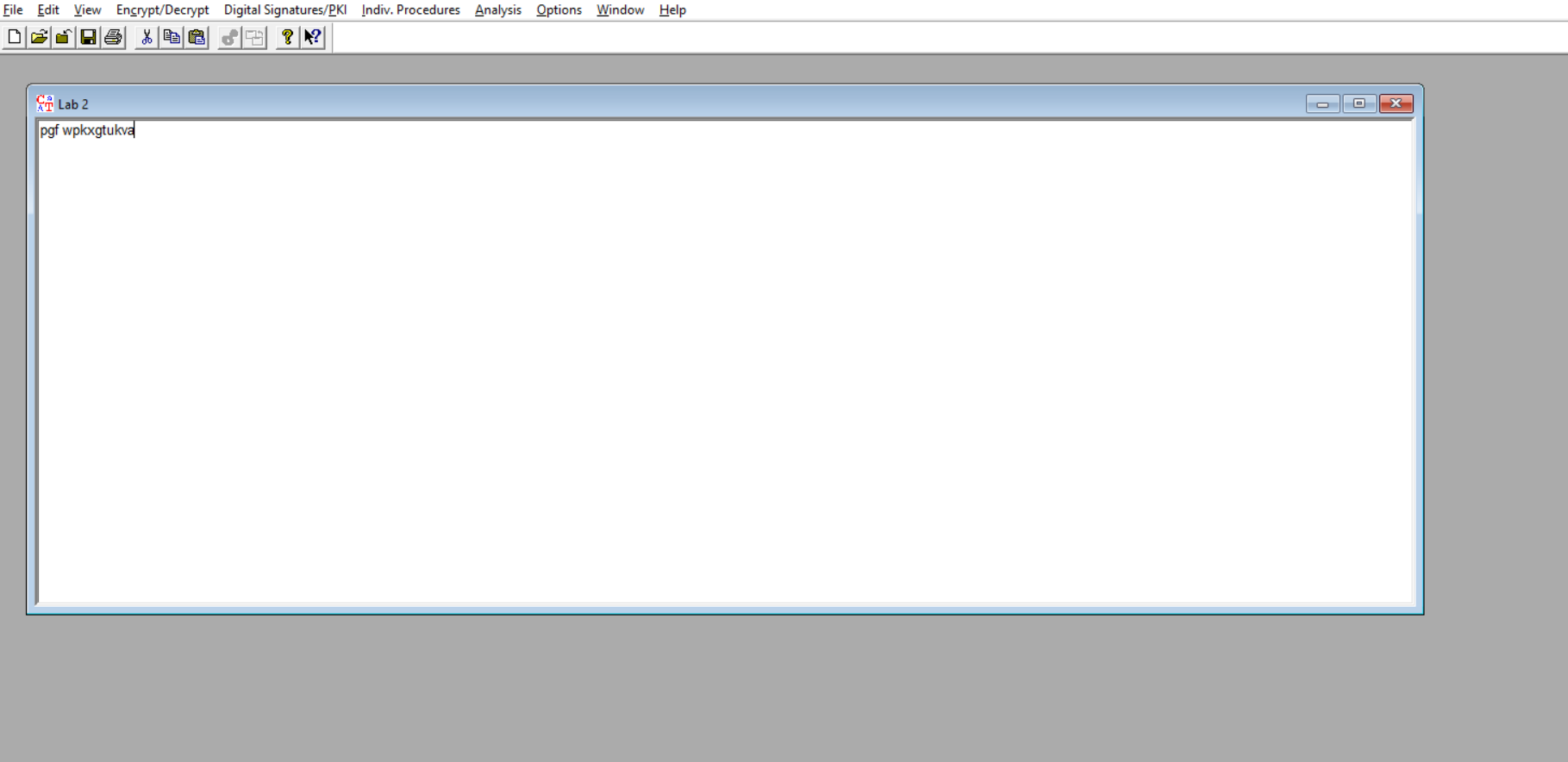
**To make a Caesar Cipher into ROT-13, you simply set the shift value to 13. Here's how it works:**

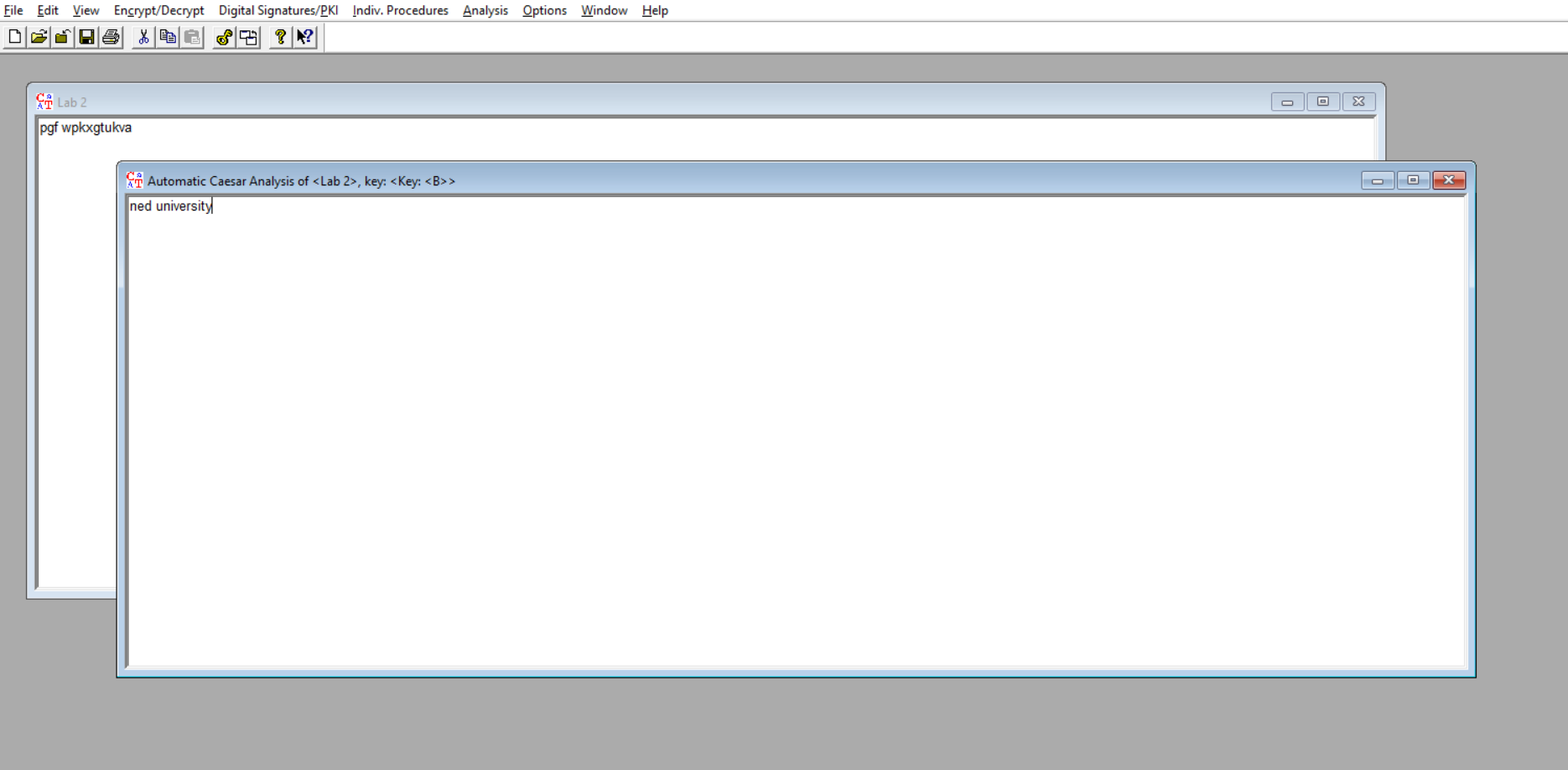
1. Caesar Cipher lets you choose any shift value (like 1, 3, or 7).
2. ROT-13 is a special case where the shift is always 13.

**For example:**

* In a Caesar Cipher with a shift of 3, A becomes D.
* In a Caesar Cipher with a shift of 13 (same as ROT-13), A becomes N.

**8. Break the encrypted text "pgf wpkxgtukva", what is the key?**

****

****

**9. Why rot-13 is considered an inverse of itself? Encrypt the same text twice using the same key and see the result?**

**Why ROT-13 is Considered an Inverse of Itself**

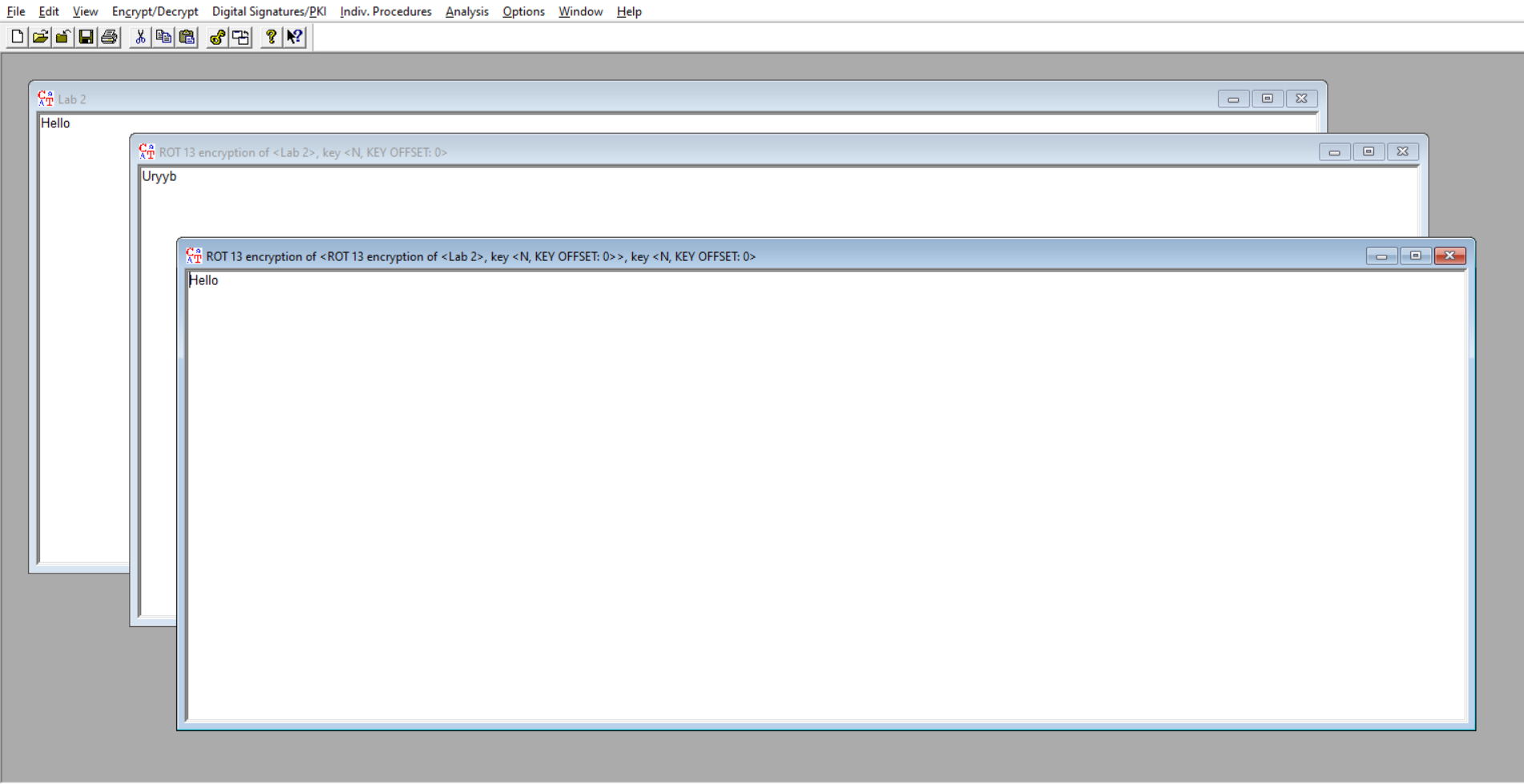
ROT-13 is a special case of the Caesar Cipher with a shift of 13. Since the alphabet has 26 letters, shifting by 13 positions forward and then another 13 positions brings you back to the original letter. This makes ROT-13 its own inverse—applying it twice returns the original text.

**Example:**

**Original Text:** HELLO

**Apply ROT-13:** URYYB

**Apply ROT-13 Again**: HELLO (back to the original)



**10. What does Frequency analysis tell us about encryption algorithm knowing that some letters are more common in the English Language than others? How can this be used in cryptanalysis? (Take the default text in Cryptool and run analysis -> tools -> histogram)**

Frequency analysis is a method used in cryptanalysis (breaking codes) that takes advantage of the fact that certain letters in a language, like English, appear more frequently than others. For example, in English, the letter **'E'** is the most common, followed by **'T', 'A', 'O', 'I',** and so on.

When a message is encrypted using simple substitution (where each letter in the plaintext is replaced by a different letter), the letter frequencies in the encrypted message (ciphertext) will still reflect the frequencies of the original letters.

**Example using "good luck":**

Let’s say the plaintext "good luck" is encrypted using a simple substitution cipher.

**Plaintext frequencies:**

* **'g'** appears once.
* **'o'** appears twice.
* **'d'**, 'l', 'u', 'c', and 'k' each appear once.

So, in the word "good luck", 'o' is the most frequent letter.

**How it's used in cryptanalysis:**

**Substitution Ciphers:** Frequency analysis is particularly effective against simple substitution ciphers because the letter frequencies of the cipher text mirror those of the plaintext.

**Monoalphabetic Ciphers:** These ciphers (where each letter is substituted by another single letter) are vulnerable since the frequency distribution remains intact.

**Polyalphabetic Ciphers:** Frequency analysis is less effective here because the substitution changes throughout the message, which alters the letter frequency distribution.

