**Permutation Cipher**

**Statement:**

Create a project with the following features:

(i) A graphical interface. The user will be given the possibility to choose the characters of the

alphabet to be used out of the blank and the 26 letters of the English alphabet. The implicit

alphabet will have all the 27 characters.

(ii) Given an encryption key and a plaintext, encrypts the plaintext. There will be a key validation

and a plaintext validation.

(iii) Given an encryption key and a ciphertext, computes the decryption key and then decrypts the

ciphertext.

**How does it work:**

If the Custom check box is not selected, then standard English alphabet is used. Otherwise the user can enter an own alphabet, with characters separated by space. The key must consist of all the numbers from 1 to desired one, all separated by space (eg: “3 2 1”). Failing to comply will lead to a warning/error dialog.

After the above fields have been filled, the user can enter the plain text to be crypted. If the user doesn't respect the alphabet, the application will send a warning message. When all the steps are completed, the user can press the Encrypt button and the encrypted message will be shown.

To decrypt the text, the user can simply select the Decrypt button.

**Main algorithms and functions:**

**split**: splits a given string into a list of strings that match a given length. if the last string does not match the length, it will receive empty characters until the lentgh is matched.

**validateKey**: makes sure that a given key is a viable permutation (no repeating numbers, all numbers from 1 to the largest)

**validateText**: converts the alphabet and plaintext to upper case and checks if the characters from the plaintext are contained by the alphabet

**encrypt**: receives a list of strings, all having the length of the key, and for each one of them creates a new string with the permuted characters. Afterwards, it constructs a result string by concatenating all the small ones.

**invert**: receives a key (as a vector of strings) and computes the inverted permutation so that it can be used with the “encrypt” algorithm with the purpose of decrypting a text