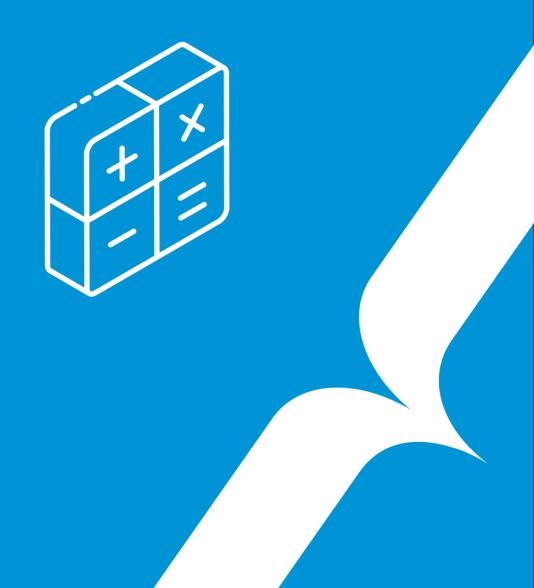
{EPITECH}

103CIPHER

MATHEMATICAL MESSAGE MASKING MULTIPLYING MATRICES



103CIPHER



binary name: 103cipher

language: everything working on "the dump"

compilation: when necessary, via Makefile, including re, clean and fclean rules



- ✓ The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- ✓ All the bonus files (including a potential specific Makefile) should be in a directory named bonus.
- ✓ Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

Cryptography is a very old science, whose goal is to secure communication, so that only its recipient could read it.

There are a lot of methods to encrypt a message, from the simplest (such as the 2,000-year-old Caesar cipher) to the most complex (such as the World War 2 Enigma code); they all need both encryption and decryption keys (sometimes identical).

In some cases (such as the Hill cipher), the key is represented by a matrix.



You have to carry out such a matrix-based ciphering software, using the following process to encrypt:

- ✓ Transcript the key into numbers using the ASCII table,
- ✓ Convert the numbered key into a square matrix, the smallest possible size, and filling the lines first.
- ✓ Transcript the clear message into numbers using the ASCII table,
- ✓ Convert the numbered message into a matrix; its number of columns should fit the key matrix size, and its number of lines should be as small as possible,
- ✓ Multiply the 2 matrices, and write the answer linearly to get the encrypted message.



During the conversion into matrices, zeros can be added at the end of the message or the key to fit the proper matrix size.

The decryption process logically follows from the previous encryption method, using the same key (be careful! You need to inverse the key matrix, which is not always possible).



2-dimension and 3-dimension matrices invertions are rather easy, but inverting bigger matrices is a difficult problem; it will be considered as a bonus if you handle it!



Usage



Using a matrix calculus library (such as numpy) is prohibited!

Suggested bonuses

- ✓ Cryptanalysis of the code, to find the original message without the key,
- ✓ Refining the encryption process.



Examples

Indeed, "Homer", transcripted into numbers using the ASCII table, gives the following 3-3-matrix:

$$\begin{pmatrix} 72 & 111 & 109 \\ 101 & 114 & 32 \\ 83 & 0 & 0 \end{pmatrix}$$

Using the ASCII table, the clear message becomes:

 $74\ 117\ 115\ 116\ 32\ 98\ 101\ 99\ 97\ 117\ 115\ 101\ 32\ 73\ 32\ 100\ 111\ 110\ 39\ 116\ 32\ 99\ 97\ 114\ 101\ 32\ 100\ 111\ 101\ 115\ 110\ 39\ 116\ 32\ 109\ 101\ 97\ 110\ 32\ 73\ 32\ 100\ 111\ 110\ 39\ 116\ 32\ 117\ 110\ 100\ 101\ 114\ 115\ 116\ 97\ 110\ 100\ 46$

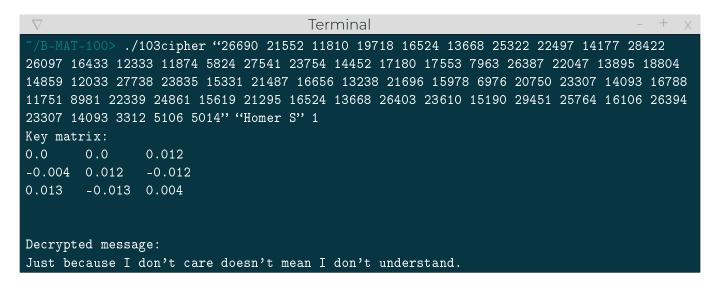
Which can be written as a 3-column-matrix:

$$74$$
 117 115
 116 32 98
 101 99 97
 \vdots \vdots \vdots
 46 0 0)



The product of these matrices is:

Which gives the encrypted message.





Elements of the key matrix are separated by tabulations in the final output.



For decryption, the key matrix is given as an indication, but will not be tested; do not bother having the exact same outuput!

