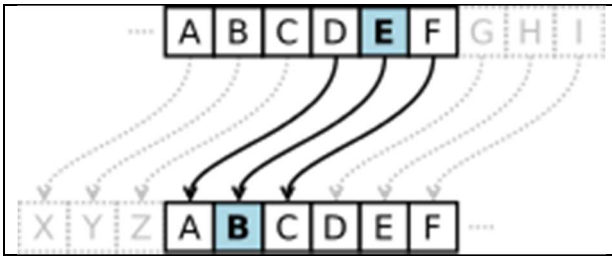
Computer Architectures **02GOLOV**	Delivery date: 3/12/2020
Laboratory **7**	Expected delivery of lab_07.zip must include: - the startup.s files for exercises 1 and 2 - this document compiled possibly in pdf format.

Starting from the *ASM_template* project, solve the following 2 exercises.

Exercise 1) In Cryptography, a **Caesar cipher** is one of the simplest encryption techniques. Using this cipher, each letter in the **plaintext** is replaced by a letter some fixed number of positions down the alphabet. The encryption **key** is the number of positions to be added to the plaintext (and subtracted from the ciphertext), in modular form (after Z, you can continue with A).
The transformation can be represented by aligning two alphabets; the cipher alphabet is the plain alphabet rotated left or right by some number of positions.




```
Plain:    ABCDEFGHIJKLMNOPQRSTUVWXYZ
Cipher:   XYZABCDEFGHIJKLMNOPQRSTUVW
```

Frequency analysis can be used to break classical ciphers. This analysis is based on the fact that, in any given stretch of written language, certain letters and combinations of letters occur with varying frequencies. Moreover, there is a characteristic distribution of letters that is roughly the same for almost all samples of that language. Considering the Italian language, the letters with the highest frequencies are the following:
E: 11.79% A: 11.74% I: 11.28% O: 9.83% N: 6.88%

An encrypted message can be memorized as a string of bytes terminated by NULL (or '\0') in the code section (as a part of the code itself) or in a read-only data section, as in the following example:

```
Ciphertext DCB "PBOAEOXDKXNYSVMYBCYNSKBMRSDODDEB",
           DCB "OOCSCDOWSNSOVKLYBKJSYXORYMKZSDYM",
           DCB "ROSVMSPBKBSYNSMOCKBOCSBYWZOPKMSV",
           DCB "WOXDOZOBAEOCDYWYDSFYOFSDOBYNSECK",
           DCB "BVYZOBZBYDOQQOBOSWSOSNKDSCOXCSLS",
           DCB "VS", 0
```

Considering the example above, write an assembly program that is able to identify the **most frequent letter** in the message. Assume that the letters are all uppercase, no whitespace, commas or numbers. Please also respond to questions in the following box.

Report your reasoning for both questions

Q1: Can you guess the encryption key by comparing the most frequent letter in your message and in the Italian language?
Because of the length of the message, it's almost certainly possible to extract the encryption key according to the fact that the most frequent letter in the clear phrase is the same letter in the encrypted one except for a difference, and this difference is the encryption key. In this example there are two most frequent letters and they are "O" and "S". The second one isn't correct while the first one is correct and the distance between "O" and "E" is 10 (16 if the decryption is done in the opposite sign).

Q2: Can you use the same strategy to find the key if the message is composed by the first 32 characters? And with a much longer message?

With only 32 characters is not possible to decrypt successfully the message because there are too few characters. With a longer message would be possible if there are a sufficient number of E letters.

Exercise 2) Create a new project by starting from the previous exercise.

Unfortunately, I didn't read completely those lab instructions so I wrote a program that computes the key and then tries to decrypt the message. In order to fill correctly the table below, only the __main procedure is taken under evaluation. The Reset_Handler's task is to clear the RAM area, useful when a soft reset happens.

The extended program manipulates the message provided in exercise 1), by applying the decryption with the key that you have found in the previous exercise (specified in a constant named KEY). The resulting message has to be stored in a proper *read-write* area. Please note that, if they KEY that you have found in Q1 is correct, the resulting decrypted message will be a clear message in the Italian language.

Report the requested values in the table below.

	Execution time @12MHz	Code size	Data size
Exercise 1)	257.5 us	46 byte	163 (ro) + 26 (rw) byte
Exercise 2)	185.0 us	46 byte	163 (rw) byte

Respond to the following open questions.

Q3: What section have you used to store the new message? A portion of a RW Area.

Q4: Which address range is assigned to this section and which memory of the system is used? The RW Area is assigned to the RAM memory of the system that starts at address 0x1000_0000. The section occupies 189 bytes so the last used address is 0x1000_00bd.