

03-60-266
Microprocessor Programming
Assignment #3
Deadline: April 8-th, 2004, by 04:00pm

Dr. Alioune Ngom

March 19, 2004

Programming Exercise #1 Write an ASM program that reads a string and encrypts it as follows:

1. All lower-case alphabetic characters are encrypted as shown below
Input character: a b c d e f g h i j k l m n o p q r s t u v w x y z
Encrypted as: h g m i f e b k d z o t c p a n y v w l x r s u q j
2. All upper-case alphabetic characters are encrypted as shown below:
Input character: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Encrypted as: B A M I F E H G D Z O T C P K N Y V W L X R S U Q J
3. All digit characters are encrypted as shown below:
Input character: 0 1 2 3 4 5 6 7 8 9
Encrypted as: 1 0 3 2 5 4 9 8 7 6
4. The following non-alphabetic non-digit characters are encrypted as shown below
Input character: ? > < & % # @ * () [] - + !
Encrypted as: % < > # ? & * ! [] () + @ -
5. All other characters are copied into the output string with no modification.

For example, the input string

```
COMP: 03-60-266 => A Computer * Organization@
```

is encrypted as

```
MKCN: 12+91+399 =< B Mkcnxlfv ! Kvbhpdjhldap*
```

After processing an input string (i.e., after displaying the output string), your program should query the user whether s/he wants to enter another string. If the answer is 'yes' (you can use Y/N for 'yes' or 'no'), read and process another string. The program terminates only when the user answers 'no' to the query.

Modular Programming: For this question, all your procedures should be in separate files. Your main program should be in file `ass3-1.asm` (this file should contain no other procedure definition). Use separate assembly to assemble all your asm files into `ass3-1.exe`.

Important: Spend some time to come up with a solution strategy as you are expected to produce an efficient code (carries 6 marks), See the marking scheme below.

Note: For this encryption, if you encrypt an encrypted string, you get back the original string. Use this feature to test your program thoroughly. Make sure to submit enough evidence that indicates that you have tested your program thoroughly.

Test: Your program will be tested on, at least, the following two strings:

COMP: 03-60-266 => A Computer * Organization@

and

AZ@[az‘} +09!

Marking Scheme: 30 points divided as follows

- Program correctness: 15 marks
- Modularity: 10 marks (modularity, ...)
- Good and efficient solutions: 5 marks (weird or convoluted or inefficient solutions will be penalized)

Programming Exercise #2 The Fibonacci function is defined as follows:

$$\text{Fibo}(n) = \text{Fibo}(n - 2) + \text{Fibo}(n - 1)$$

Given that $\text{Fibo}(0) = \text{Fibo}(1) = 1$, write an ASM program that generates a Fibonacci sequence of length l , that is the sequence

$$\text{Fibo}(0), \text{Fibo}(1), \text{Fibo}(2), \dots, \text{Fibo}(l - 2), \text{Fibo}(l - 1), \text{Fibo}(l)$$

Your main program, called `ass3-2.asm`, should call the procedure named `Sequence`, with parameter l , to generate a sequence of length l . The procedure `Sequence` should call the procedure `Fibonacci` to compute the Fibonacci value of a given number n . The main program and the 2 procedures should be in distinct files: `ass3-2.asm`, `Sequence.asm`, and `Fibonacci.asm`. Also, feel free to define other procedures to help break down the complexity of your implementation (any defined procedure should be in a distinct file). Use modular programming and separate assembly, for this question.

Marking Scheme: 30 points, same as in question #1.