30 January 2013 -- Computer Architectures –- part 1/2

Last and First Name, Matr. ....

One of the simplest algorithms for encrypting information is the Caesar code. Here we consider variations of that algorithm. Let us consider the case where we have to encrypt short messages, up to 140 characters long, using capital letters (from A to Z) ONLY and NOTHING ELSE. Let us consider a seed denoted with **λ** such that $0 <= \lambda < 30$. Let us denote with symbols **α_i** and **μ_i** the clear and encrypted characters of the message, respectively with i being the character position in both the "clear" and in the "encrypted" text. For sake of simplicity, in the following **α_i** represents the ascii code of the "clear" text.
Let us consider now two encryption algorithms: Algo_E and Algo_D, using simple operations modulo β (see later).

Algo_E encryption	Algo_D encryption
Receive λ	Receive λ
Encrypt first char α_1 into $\mu_1 = \mathrm{mod}_\beta(\alpha_1 + \lambda)$	Encrypt first char α_1 into $\mu_1 = \mathrm{mod}_\beta(\alpha_1 + \lambda)$
Encrypt second char α_2 into $\mu_2 = \mathrm{mod}_\beta(\alpha_2 + \mu_1)$	Encrypt second char α_2 into $\mu_2 = \mathrm{mod}_\beta(\alpha_2 + \mu_1 + \lambda)$
Encrypt third char α_3 into $\mu_3 = \mathrm{mod}_\beta(\alpha_3 + \mu_2)$	Encrypt third char α_3 into $\mu_3 = \mathrm{mod}_\beta(\alpha_3 + \mu_2 + \mu_1 + \lambda)$
Continue encrypting	Continue encrypting
Encrypt last char α_i into $\mu_i = \mathrm{mod}_\beta(\alpha_i + \mu_{i-1})$	Encrypt last char α_i into $\mu_i = \mathrm{mod}_\beta(\alpha_i + \mu_{i-1} + \ldots + \mu_2 + \mu_1 + \lambda)$

Decryption operates in the reverse way (i.e. modulo subtractions), from first to last character. It is required to write a 8086 assembly program working under the following assumptions and definitions:

```
CLEAR DB N DUP (?)      ; is the array of the N characters (N from 5 to 140) to be encrypted in its full length
LAMBDA DB ?             ; is the seed, with 0<= LAMBDA < 30 received in input by the program
ENC_E DB N DUP (?)      ; is the array of the encrypted text, according to ALGO_E
ENC_D DB N DUP (?)      ; is the array of the encrypted text, according to ALGO_D
DEC_E DB N DUP (?)      ; is the array of the decrypted text, according to ALGO_E and array ENC_E
DEC_D DB N DUP (?)      ; is the array of the decrypted text, according to ALGO_D and array ENC_D
I  DB  ?                ; position of a character either in the encrypted or decrypted array
DX_I DB  ?              ; decrypted character in position I
```

to compute the following:

- **Item 1**: In the special case **β=128**, and having received λ and CLEAR in input by the user at the beginning of the program, computes ENC_E and prints its codes (in the form of numbers) on the screen. It is forbidden to use the operations of division of the 8086 assembly language.
- **Item 2**: In the special case **β=128**, and having received λ and ENC_E in input by the user at the beginning of the program (the latter as a sequence of numbers), computes DEC_E and prints its contents (in the form of the message) on the screen. It is forbidden to use the operations of division of the 8086 assembly language.
- **Item 3**: In the special case **β=128**, and having received the position I, λ and ENC_E, in input by the user at the beginning of the program (the latter as a sequence of numbers), computes DX_I without fully decrypting the previous characters and prints its contents (in the ascii form) on the screen. It is forbidden to use the operations of division of the 8086 assembly language.
- **Item 4**: In the special case **β=128**, and having received λ and CLEAR in input by the user at the beginning of the program, computes ENC_D and prints its codes (in the form of numbers) on the screen. It is forbidden to use the operations of division of the 8086 assembly language.
- **Item 5**: In the special case **β=128**, and having received λ and ENC_D in input by the user at the beginning of the program (the latter as a sequence of numbers), computes DEC_E and prints its contents (in the form of the message) on the screen. It is forbidden to use the operations of division of the 8086 assembly language.
- **Item 6**: In the special case **β=128**, and having received the position I, λ and ENC_D, in input by the user at the beginning of the program (the latter as a sequence of numbers), computes DX_I without fully decrypting the previous characters and prints its contents (in the ascii form) on the screen. It is forbidden to use the operations of division of the 8086 assembly language.

****

- **Item 7**: In the case **β=61**, and having received λ and CLEAR in input by the user at the beginning of the program, computes ENC_D and prints its codes (in the form of numbers) on the screen, without using, in an efficient way, the operations of division of the 8086 assembly language.

30 January 2013 -- Computer Architectures -- part 1/2

Last and First Name, Matr. ....

- **Item 8**: In the case **β=61**, and having received λ and ENC_D in input by the user at the beginning of the program (the latter as a sequence of numbers), computes DEC_D and prints its contents (in the form of the message) on the screen, without using, in an efficient way, the operations of division of the 8086 assembly language.
- **Item 9**: In the case **β=61**, and having received the position I, λ and ENC_D, in input by the user at the beginning of the program (the latter as a sequence of numbers), computes DX_I without fully decrypting the previous characters and prints its contents (in the ascii form) on the screen, without using, in an efficient way, the operations of division of the 8086 assembly language.

Items with corresponding points for each completed item (as uncompleted items could be not evaluated):
Items 1-6 are NOT COMPATIBLE with Items 7-9; please solve only inside Items 1-6 or only inside Items 7-9

- Item 1. OR Item 2.: up to **7.5 points** each; (i.e. encrypt and decrypt with Algo_E and **β=128**, will bring 7 points each)
- Item 3.: up to **1 point**;
- (Item 1. + Item 4.) OR (Item 2. + Item 5.): up to **12 points** each pair; (i.e. encrypt with Algo_E together with Algo_D again, **β=128**, will bring 12 points; same also for decrypt)
- Item 4., Item 5.: up to 10 point each (i.e. encrypt and decrypt with Algo_D and **β=128**, will bring **10 points** each, **if no Item 1./Item 2. has already been solved**).
- Item 6.: up to **2 points**

****

- Item 7. OR Item 8.: up to **15 points** each;
- Item 9: up to **3 points**

EXAMPLES. Assume CLEAR = "ABCD" and λ = 20 (the ascii code of "A" is 65, of "B" is 66 and so on)

Algo_E encryption (**β=128**) Item 1.; decry. is rev.	Algo_D encryption (**β=128**) Item 4. decryption is reverse
$\mu_1 = mod_{128}(65+20) = 85$	$\mu_1 = mod_{128}(\alpha_1 + \lambda) = mod_{128}(65+20) = 85$
$\mu_2 = mod_{128}(66+85) = mod_{128}(151) = 23$	$\mu_2 = mod_{128}(\alpha_2 + \mu_1 + \lambda) = mod_{128}(66+85+20) = mod_{128}(171) = 43$
$\mu_3 = mod_{128}(67+23) = 90$	$\mu_3 = mod_{128}(\alpha_3 + \mu_2 + \mu_1 + \lambda) = mod_{128}(67+43+85+20) = 87$
$\mu_4 = mod_{128}(68+90) = mod_{128}(158) = 30$	$\mu_4 = mod_{128}(\alpha_4+\mu_3+\mu_2+\mu_1+\lambda) = mod_{128}(68+87+43+85+20) = 47$

Algo_D encryption (**β=61**) Item 7. decryption is reverse
$\mu_1 = mod_{61}(\alpha_1 + \lambda) = mod_{61}(65+20) = 24$
$\mu_2 = mod_{61}(\alpha_2 + \mu_1 + \lambda) = mod_{61}(66+24+20) = 49$
$\mu_3 = mod_{61}(\alpha_3 + \mu_2 + \mu_1 + \lambda) = mod_{61}(67+49+24+20) = 38$
$\mu_4 = mod_{61}(\alpha_4+\mu_3+\mu_2+\mu_1+\lambda) = mod_{61}(68+38+49+24+20) = 16$

REQUIREMENTS (SHARP)

- It is not required to provide the optimal (shortest, most efficient, fastest,…) solution, but a working and clear solution.
- It is required to write at class time a short and clear explanation of the algorithm used.
- It is required to write at class time significant comments to the instructions.
- The input-output part is not necessary in the class-developed solution, but its implementation is mandatory to be discussed at oral exam.
- Minimum score to "pass" this part is 15 (to be averaged with second part and to yield a value at least 18)

Please use carbon copy ONLY (NO PICTURES ARE ALLOWED) and retain one copy for home implementation and debug. Please provide your classroom submitted solution with several explanatory and significant comments. When coming to oral discussion, please mark on your "classroom" copy, all modifications. Please also provide an error-free and running release of the solution, as well as with its printed list of instructions. Please consider that the above are necessary but not sufficient requirements to success the exam, since the final evaluation will be based on a number of parameters.
FAILURE TO ACCOMPLISH ALL PREVIOUS NECESSARY REQUIREMENTS WILL CAUSE NO-QUESTION-ASKED AND IMMEDIATE REJECTION.