Assignment 6 (pa6.asm)

NOTE: The word shift is used in the problem statement, but this does not refer to the SHIFT instructions in Assembly. Instead it refers to the Caesar Cipher.

The goal of this assignment is to create an encryption/decryption system.

A menu will allow the user to enter the encryption/decryption key and the phrase to be encrypted decrypted. The user will be asked if they are encrypting or decrypting a string. If a Key has been previously entered, ask the user if they want to use that key or enter a new one.

You will need to convert the entered phrase to be encrypted into all uppercase (ChangeCase PROC). You will also need to remove all non-letter elements (LettersOnly PROC). So that you are only encrypting/decrypting letters. The Key can be any combination of Alphanumeric characters and any symbol on the keyboard.

The encryption/decryption method is a variation of the Caesar Cipher (https://en.wikipedia.org/wiki/Caesar_cipher). The encryption/decryption key will be a word entered by the user and can be upper/lower case and contain punctuation. The ASCII value of each element of the key will be used to determine how far to shift/rotate the corresponding letter of the phrase to be encrypted to the **right**. Decrypting a word/phrase will require a shift/rotate to the **left**. Requires two proc's: encrypt PROC and decrypt PROC

You **MUST** take the modulus base $1A_{16}$ of the ASCII hex value of each letter of the key to see how far to shift each letter of the word/phrase to be encrypted/decrypted. Hint: There is no modulus instruction in ASSEMBLY, but there is something you can use instead. If you don't know what the modulus is, you can find more information at https://en.wikipedia.org/wiki/Modulo_operation.

After Encrypt/Decrypt is executed the result must be a capital letter.

The key may be shorter than the word/phrase to be encrypted/decrypted. If this is the case, the key repeats.

The encrypted/decrypted word/phrase will be printed out in the following format. 5 letters then a space then 5 letters etc. This means you CAN NOT use WRITESTRING from the Irvine Library to print out your result. See example below. (Printit PROC)

***** ***** ***** Where each * represents a letter.

The user should be presented with a menu asking what they want to do. If the user enters an empty phrase, tell them so and ask for a phrase.

You may safely assume the user will not enter a string longer than 50 characters for either the phrase or the key.

Remember to save the key somewhere so that if the user wants to immediately decrypt after encrypting, they can. The user should not be able to encrypt or decrypt until a key is entered (at least once).

An Example: The phrase to be encrypted is The Dog and the key is BONE.

Given these ASCII values from the back of the book

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
41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53	54	55	56	57	58	59	5A

Ecryption Process

The Phrase to be encrypted	T	H	E	D	O	G
Ascii code of the phrase	54	48	45	44	4F	47
The Key, repeated as necessary	B	O	N	E	B	O
Ascii code of the key	42	4F	4E	45	42	4F
Mod $1A_{16}$ of key	E	1	0	11	E	1
Ascii value of Phrase plus shift	62	49	45	55	5D	48
Encrypted Phrase	H	I	E	U	C	H

If the sum of Ascii value of phrase + shift amount is greater than the max value (5A), you must correct to ensure you get a value within the Capital letters.

Correction – See highlighted entries in table

HIEOC H

Specifications:

1. I will not be debugging your code. It must stand alone. This means I should not have to enter a call to waitmsg to see the result. I will “Start without Debugging” to test your code.

2. You may use any instruction for Chapter 7 and below. You may not use instructions not yet discussed in class.

3. All procedures and main will have the required header comment block.

4. Comments are required.

5. Do not use any . directives to assist with program flow, eg, .if, .while, .else, etc.

6. Remember your procedures should accomplish a basic task, so it may be that you write more procedures than are indicated above.

7. You must pass variables using registers. Do not use the STACK to pass variables (that’s another chapter). If you refer to a variable that is not declared in your procedure then that variable must be passed by reference to the procedure (via its offset in a register). If you want to pass data to another procedure you must use a register to pass it out of your procedure or have passed in a variable (via its offset in a register).

8. Remember that you can create local variables, but that they are scoped (just like c++). No GLOBAL variables. No referring to variables by name not declared in the procedure. The only exception to this is main PROC.

9. Submit a pdf of the encryption of “This is a Caesar Cipher Test”. Encrypted with the Key “Assembly Homework”. Also submit a pdf of the decryption of the result from the above. Use the same key. Naturally, you must also submit your code.

10. As always, style is a component of your grade.