Assignment #02

Submission Deadline:Friday,4 dec 2020,11:50PM

Total Marks: 90

Instructions

a. Programs must be Masm 615 compatible. We will not test your code on Emulator.
b. Code must be commented properly.
c. You have to submit all your codes in a **asm files;**
d. Submitted Filename must have format like **Assignment#_Sec#_Roll#** e.g. A1_A_15_i1234
e. **Plagiarism will result in ZERO Marks in all assignments of Class and Lab.**

Section:
A B C D E F G H

COAL Fall 2020

Assignment#02

Question No.1 (10)

Write an assembly language procedure named "equalsIgnoreCase", which receives two strings and their sizes, and returns true if the two strings contain the same characters irrespective of the case. For example, for strings {"aBc"} and {"Abc"} the function returns true, but for {"aBc"} and {"aB"}, or {"aBc"} and {"Xbz"}, the function returns false. Write a generic procedure that must handle all checks and conditions.

Question No.2(15)

Write a subroutine print triangle that draw a Triangle , that starts from the middle of the first row of screen having Height H. H should be less than 25. There are 160 bytes in each row of dosbox screen.

```
mov ax,03h        ;; clear screen
int 10h           ;; and sets cursor to the top
```

In each iteration move that triangle towards the bottom of the screen (now starts from the next line) after some delay. When the base of triangle touches the bottom of screen , terminate your program. For the above problem, a program is given to you which have a delay subroutine, a clear screen subroutine, a print_triangle subroutine and a main program. Everything is assembled and written; you have to write the print_triangle subroutine, and also complete the terminating condition of the main program. You can use asterisk as for printing. Asterisk hex value is 2A.

Input the value of H from the user.

Delay subroutine is given with assignment document name as "delay_code"

Question No .3(15)

Perfect Numbers:

An integer is said to be a perfect number if the sum of its factors, including 1 (but not the number itself), is equal to the number. For instance, 6 is a perfect number (**6 = 1 + 2 + 3**).Write an assembly language program to check whether an input number is perfect or not. Your main procedure should ask the user to input a number. Input-Number should be between 1 and 100. Now pass this number to a procedure named "perfect". Procedure 'perfect' should determine whether the parameter is a perfect number or not. It should also print all factors of a perfect number. Use stack addressing to store factors of a perfect number. You are not allowed to use array to store factors of a perfect number

Section:
A B C D E F G H

COAL Fall 2020

Assignment#02

Question No .4 (25)

A problem in elementary algebra is to decide if an expression containing several kinds of brackets, such as, [,],{,},(,), is correctly bracketed. This is the case if (a) there are the same number of left 'and right brackets of each kind, and (b) when a right bracket appears, the most recent preceding unmatched left bracket should be of the same type. For example,·

(a + [b – {c × (d – e) }] + f) is correctly bracketed, but

(a + [b – {c × (d – e)] } + f) is not

Correct bracketing can be decided by using a stack. The expression is scanned left to right. When a left bracket is encountered, it is pushed onto -the stack. When a right bracket is encountered, the stack is popped (if the stack is empty, there are too many right brackets) and the brackets are compared. If they are of the same type, the scanning continues. If there is a mismatch, the expression is incorrectly bracketed. At the end of the expression, if the stack is empty the expression is correctly bracketed. If the stack Is not empty, there are too many left brackets.

Write a program that lets the user type in an algebraic expression, ending with a carriage return, that contains round (parentheses), square, and curly brackets. As the expression is being typed in, the program evaluates each character. If at any point the expression is incorrectly bracketed (too many right brackets or a mismatch between left and right brackets), the program tells the user to start over. After the carriage return is typed, If the expression is correct, the program displays "expression Is correct." If not, the program displays "too many left brackets". In both cases, the program asks the user if he or she wants to continue. If the user types 'Y', the program runs again. Your program does not need to store the input string ,only check it for correctness

Sample execution:

```
ENTER AN ALGEBRAIC EXPRESSION:
(a + b)]TOO MANY RIGHT BRACKETS. BEGIN AGAIN!
ENTER AN ALGEBRAIC EXPRESSION
(a + [b - c] x d)
EXPRESSION IS CORRECT
TYPE Y IF YOU WANT TO CONTINUE:Y
ENTER AN ALGEBRAIC EXPRESSION:
[a + b x (c - d) - e}BRACKET MISMATCH. BEGIN AGAIN!
ENTER AN ALGEBRAIC EXPRESSION:
((a + [b - {c x (d - e) } ] + f)
TOO MANY LEFT BRACKETS. BEGIN AGAIN!
ENTER AN ALGEBRAIC EXPRESSION:
I'VE HAD ENOUGH
EXPRESSION IS CORRECT
TYPE Y IF YOU WANT TO CONTINUE:N
```

Question No.5 (10+15)

Create a procedure that generates a four-by-four matrix of randomly chosen capital letters. When choosing the letters, there must be a 50% probability that the chosen letter is a vowel. Write a test program with a loop that calls your procedure five times and displays each matrix in the console window. Following is sample output for the first three matrices:

D W A L
S I V W
U I O L
L A I I

K X S V
N U U O
O R Q O
A U U T

P O A Z
A E A U
G K A E
I A G D

Part 2

Use the letter matrix generated in the previous question as a starting point for this program. Generate a random four-by-four letter matrix in which each letter has a 50% probability of being a vowel. Traverse each matrix row, column, and diagonal, generating sets of letters. Display only four-letter sets containing exactly two vowels. Suppose, for example, the following matrix was generated:

P O A Z
A E A U
G K A E
I A G D

Then the four-letter sets displayed by the program would be

Output

POAZ, GKAE, IAGD, PAGI,
ZUED, PEAD, and ZAKI.
The order of letters within each set is unimportant.

~Best of Luck~ ☺