60-266 – Assignment #2

DUE DATE is: *Friday, February 16, 2018.* To be submitted via Blackboard by Midnight.

WARNINGS: You must only use instructions and directives discussed from Lecture 1 (Chapt_01.pptx) to Lecture 6 (Chapt_04-c.pptx).

Programming Exercise 1 (100 points): [call it Ass2-Q1.asm]

Write an ASM program which:

1. Fills in an array `Vector` with at most 50 signed double-word integers from the keyboard;
2. Computes the sum of all the negative values in Vector;
3. Counts the number of all the positive values in Vector;
4. Finds the minimum value between position I and position J of Vector;
5. Checks whether Vector is a palindrome or not

Your program will display the following interaction with you (things in reds are your inputs)

```
➢  What is the size N of Vector?> 13
➢  What are the 13 values in Vector?> -1  +3  +17  0  -100  -30  +2  -30  -100  0  +17  +3  -1
➢
➢  Size of Vector is N = 13
➢  Vector = -1    +3    +17    0    -100    -30    +2    -30    -100    0    +17    +3    -1
➢
➢  The sum of all the negative values in Vector is: Sum = -262
➢  The number of all the positive values in Vector is: Count = 5
➢
➢  Please give me two values I and J such that 1 ≤ I ≤ J ≤ N> 2 7
➢
➢  I = 2 and J = 7, and
➢  The minimum value between position 2 and 7 of Vector is: Minimum = -100
➢
➢  Vector is a palindrome because it reads the same way in both directions.
➢
➢  Repeat with a new Vector of different size and/or content? > N
```

a. If `N` is negative, the program should display "`Size must be positive or zero`" and prompts you for `N` again.

b. If `I < 1`, or `J < 1` or `I`, or `N < 1` or `I` or `J`, the program should display "`Invalid I or J`" and prompts you for `I` and `J` again.

c. If Vector is not a palindrome, the program should display "`Vector is NOT a palindrome`".

d. The program repeats again from the first prompt if you type `Y` to the last question, otherwise it exits.

Make use of data-related operators as much as possible, such as OFFSET, SIZEOF, TYPE, LENGTHOF, DUP or PTR, etc, in order to make your program as flexible as possible (and as short/efficient as possible); see Chapt_04-c.