Homework 6 (100 points)
Due Date*: 10:00am 04/02/2018, Cutoff Deadline: 10:00am 04/05/2018**
****Submission will NOT be accepted after the cutoff deadline**
Submission: The .s file on Blackboard under Homework 6 (NO email submission please!)

 **Warning**: (1) The grading of an ARM program is based on the testing results. An ARM program that cannot be assembled will receive up to 5% of the points!!! (2) Once the submitted work is graded, no more submission will be accepted or re-graded.

Write an ARM program to **correct** and **decode** an **even-parity Hamming code** into a **source word**

- in the data area,
  - define a symbol called **MAX_LEN** and equivalent it with a number like 100
  - declare and initialize a ***NULL-terminated string*** as a **Hamming code**, label it as **HCode,** and initialize it using the value of your choice (each byte is the ASCII of a bit in your Hamming code). For example, the string could be "11111100**0**001101", 0x0 for ***case 1*** below.
  - reserve a ***chunk of zeroed memory*** with a length of **MAX_LEN** bytes, label this chunk of memory as **SrcWord**, which will be used to store a Null-terminated string representing the corrected **source word**.
  - Store each one of the above four labels as a word with an address label, e.g., "adrHCode DCD HCode" for HCode, and **EXPORT** each address label, e.g., "EXPORT adrHCode".
- in the main program,
  - go through the Hamming code string via examining EACH bit (data bit and check bit), count the total number of 1's in EACH check bit's parity and update the relevant counters for the check-bits.
  - go through the counters to find all the **check bits** whose **parities** are NOT **even**, and add together the **sequence numbers** of all those **check bits** whose **parities** are NOT **even**.
  - use the above sum of the **sequence numbers** to find THE INCORRECT bit if ANY in Hamming code , and **invert** this bit
  - go through the CORRECTED **hamming code** to retrieve the **source word**, and store the source word at SRC_WORD as a NULL-terminated string.

Suggested Test cases for your program:

Case 1: use "11111100**0**001101" as the Hamming code, your program is going to detect that bit #9 in this Hamming code is transmitted wrong, correct it as '**1**', and extract the source word "11101001101" as a NULL-terminated string at SRC_WORD

Case 2: use "010011100101" as the Hamming code, your program is going to detect that NO bit in this Hamming code has an even-parity-error, and extract the source word "01110101" as a NULL-terminated string at SRC_WORD

More cases: you may use HW5a posted on blackboard to find more pairs of Hamming code and source word, then

- use this Hamming code as the No-Even-Parity-Error case like case 2
- invert ONE bit in this Hamming code and use it as the Even-Parity-Error case like case 1