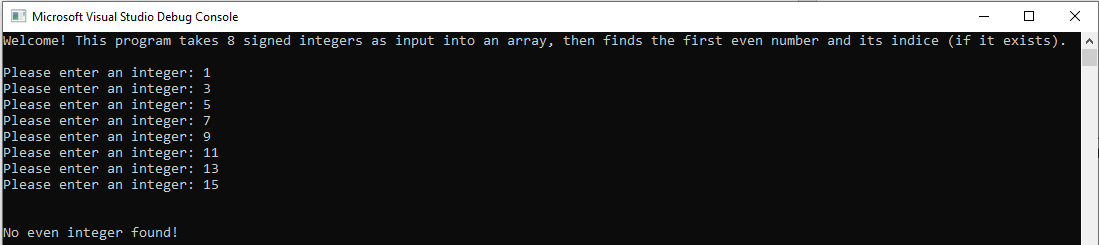
# Comp 3350: Computer Organization & Assembly Language

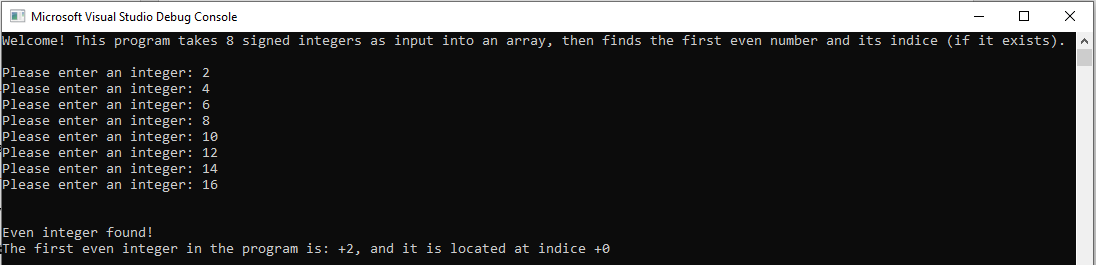
# HW # 7: Theme: Conditionals, Booleans, Loops

*(All main questions carry equal weight. Credit awarded to only those answers for which work has been shown.)*

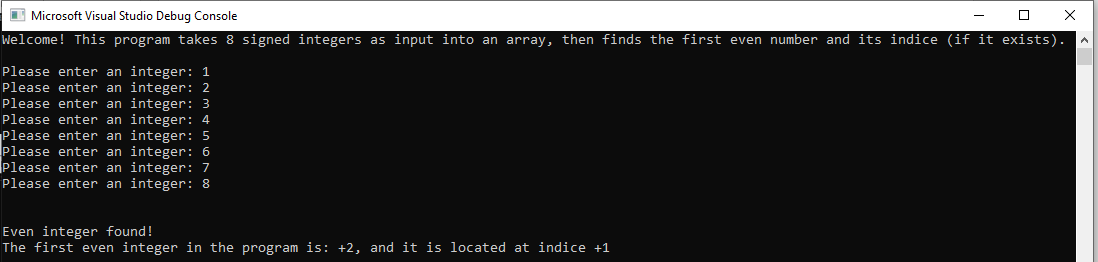
1. Draft a program that scans an array to determine the first even integer in an array. If a value is found, the program should print “even integer found” its value and index. If no even integer is found, the program should print “no even integer found.” **Submit the asm/list file and screenshots that shows the output of your code for the following example arrays:** 
   1. Array has all odd integers

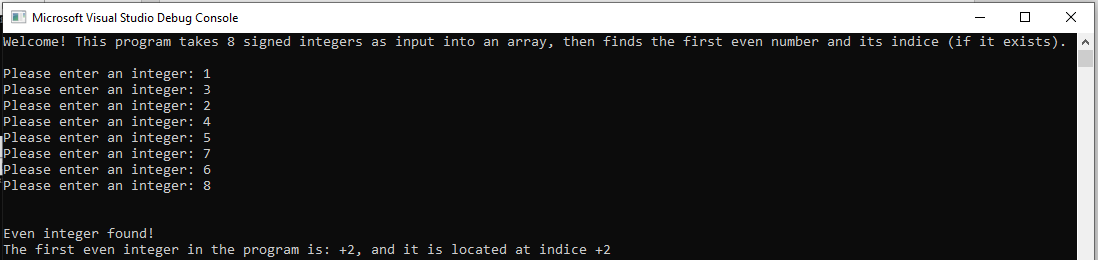


* 1. Array has all even integers



* 1. Several arrays with a mix of odd and even integers positioned at different indices





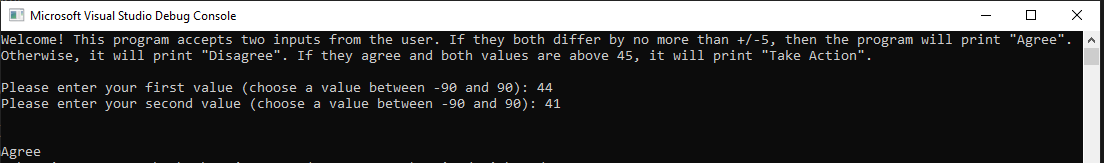
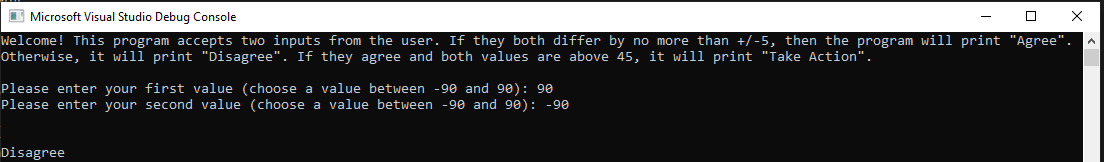
1. Write a program which encodes any string using the XOR instruction. Test it using your <first name last name> in the data segment to produce cipher text and then decode using the program to get plain text. Use the last two digits of your student id as the key. Print plane text from the data segment, print the cipher text, and then print the plain text upon execution. **Submit the asm/list file and screenshots that shows the output of your code.**

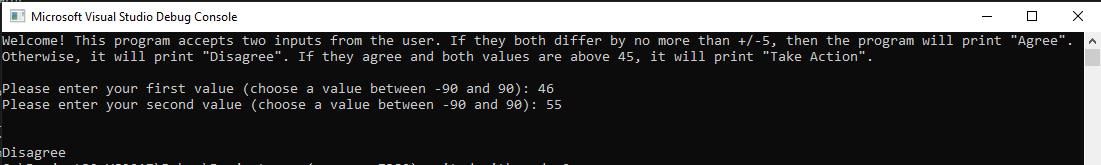
****

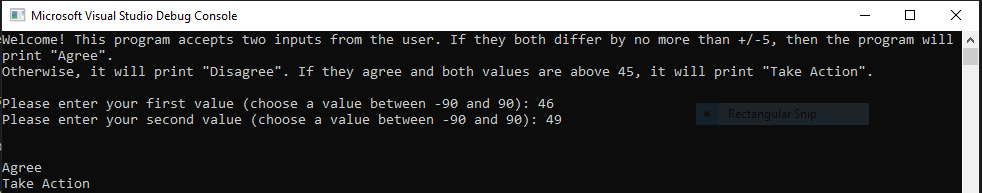
What are the strengths and weaknesses of this encryption method (**25% of points, Typewritten answer required**)?

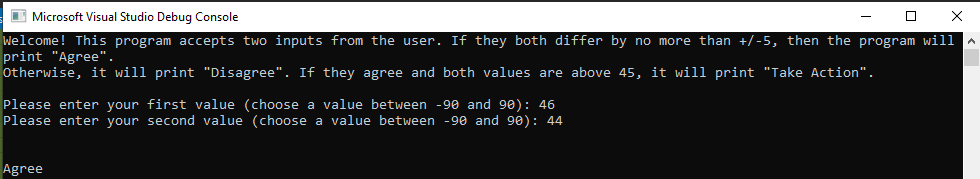
Some strengths of this encryption method is that it is easy to implement. Another strength is that it is difficult to break without the key. Some weakness of this encryption method are that you must have the key to decrypt the message. If you try and send the key then it can be intercepted, so you must determine the key beforehand and it is hard to change the key once chosen. Another weakness is that you can use character occurrence to guess which characters represent which decrypted characters, which would not take long with the right computer. Once you can decrypt one character then you can decrypt them all.

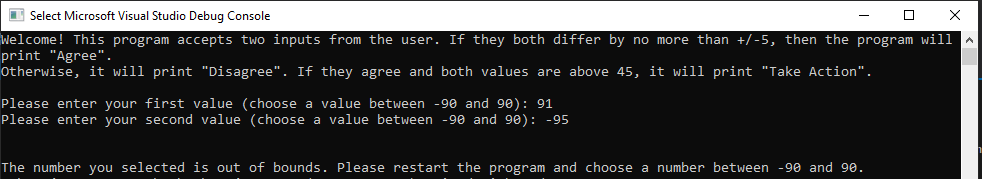
1. Write a program that gets its input from two sensors. If the values of the sensors differ by no more than +/- 5, print “Agree”, otherwise, print “Disagree.” You can assume that the values are integers. Additionally, if the values Agree and they are each more than 45, print “Take Action”. **Submit asm/list file and show screenshots of robust testing for various inputs, including boundary conditions, in the closed interval (-90 … 90).**

****

****

****

****

****

1. Draw the stack (word/pdf) before every instruction that is marked red is executed to show your understanding of the call and return functions. Use N/A to represent unpredictable values.

Main Proc

4040018 mov ecx, 0000000Ch

404001C mov ebx, 0000000Bh

4040020 call FMul

4040026 mov eax, ebx

…

…

Main EndP

FMul PROC

4041040 Push ecx

4041044 Push ebx

4041048 mov eax, edx

…

…

404A060 Pop ebx

404A062 Pop ecx

404A064 ret

FMul EndP

**Before call FMul:**

Offset:

0000 1000 N/A (ESP)

0000 0FFC N/A

0000 0FF8 N/A

0000 0FF4 N/A

EIP: 0404 0020

**Before push ecx:**

Offset:

0000 1000 4040026 (ESP)

0000 0FFC N/A

0000 0FF8 N/A

0000 0FF4 N/A

EIP: 0404 1040

**Before push ebx:**

Offset:

0000 1000 04040026

0000 0FFC 0000000C (ESP)

0000 0FF8 N/A

0000 0FF4 N/A

EIP: 0404 1044

**Before mov eax, edx:**

Offset:

0000 1000 04040026

0000 0FFC 0000000C

0000 0FF8 0000000B (ESP)

0000 0FF4 N/A

EIP: 0404 1048

**Before pop ebx:**

Offset:

0000 1000 04040026

0000 0FFC 0000000C

0000 0FF8 0000000B (ESP)

0000 0FF4 N/A

EIP: 0404 A060

**Before pop ecx:**

Offset:

0000 1000 04040026

0000 0FFC 0000000C (ESP)

0000 0FF8 0000000B

0000 0FF4 N/A

EIP: 0404 A062

**Before ret:**

Offset:

0000 1000 04040026 (ESP)

0000 0FFC 0000000C

0000 0FF8 0000000B

0000 0FF4 N/A

EIP: 0404 A064

**Before mov eax, ebx:**

Offset:

0000 1000 04040026 (ESP)

0000 0FFC 0000000C

0000 0FF8 0000000B

0000 0FF4 N/A

EIP: 0404 0026