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

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

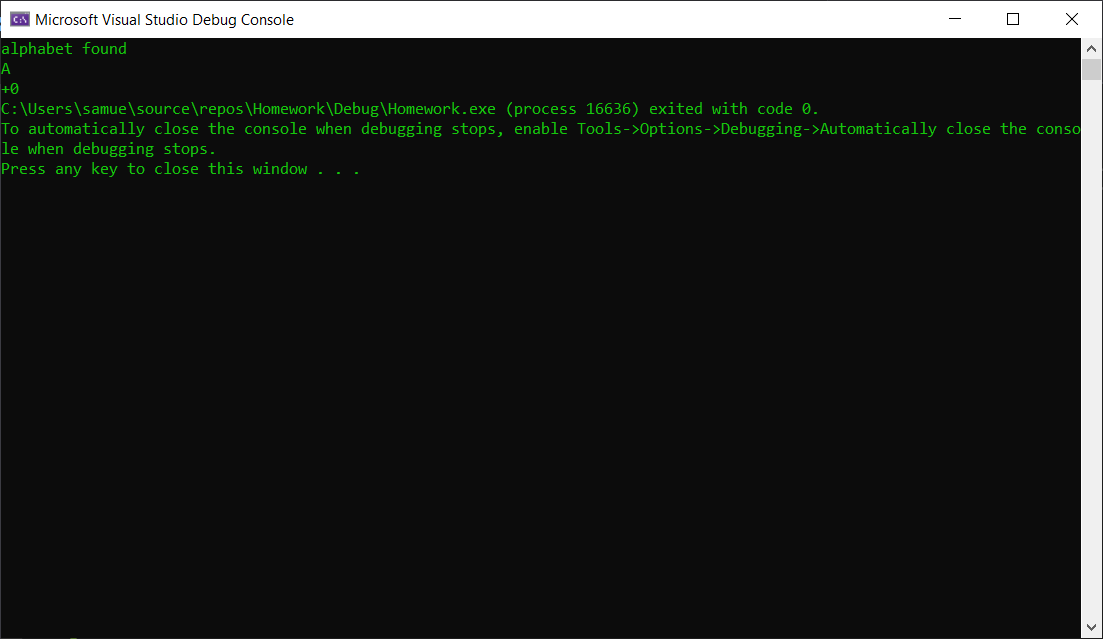
1. Draft a program that scans an alphanumeric array to determine the first alphabet in the array.  If found, the program should print “alphabet found” its value and index.  If no alphabet is found, the program should print “no alphabet found.” **Submit the asm/list file and screenshots that show the output of your code for the following example arrays:**

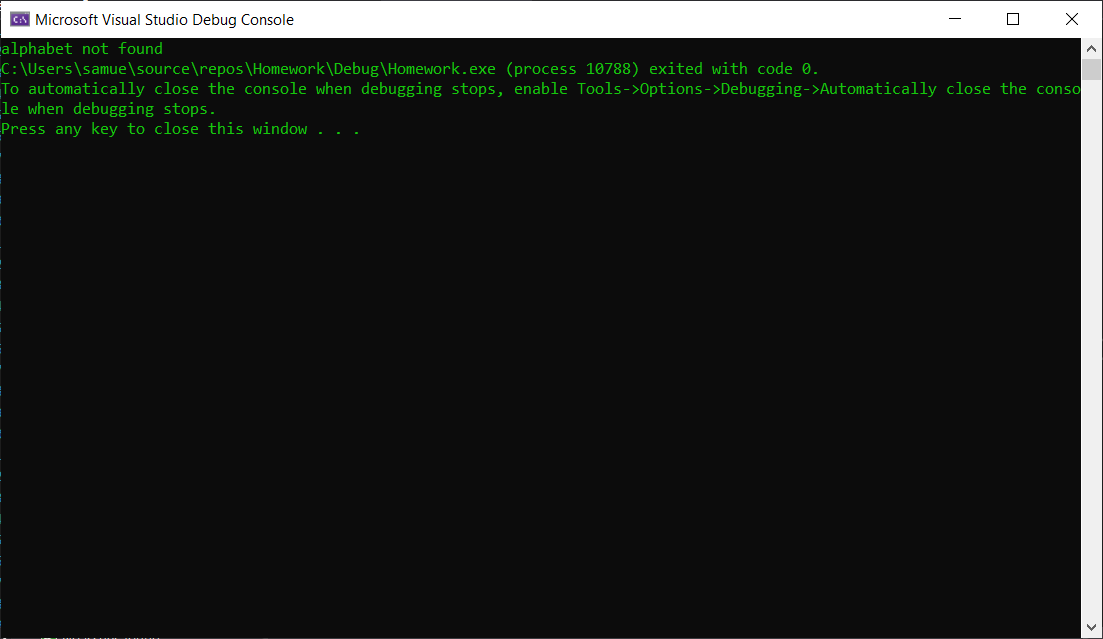
a. Array has only alphabets; Boundary Case

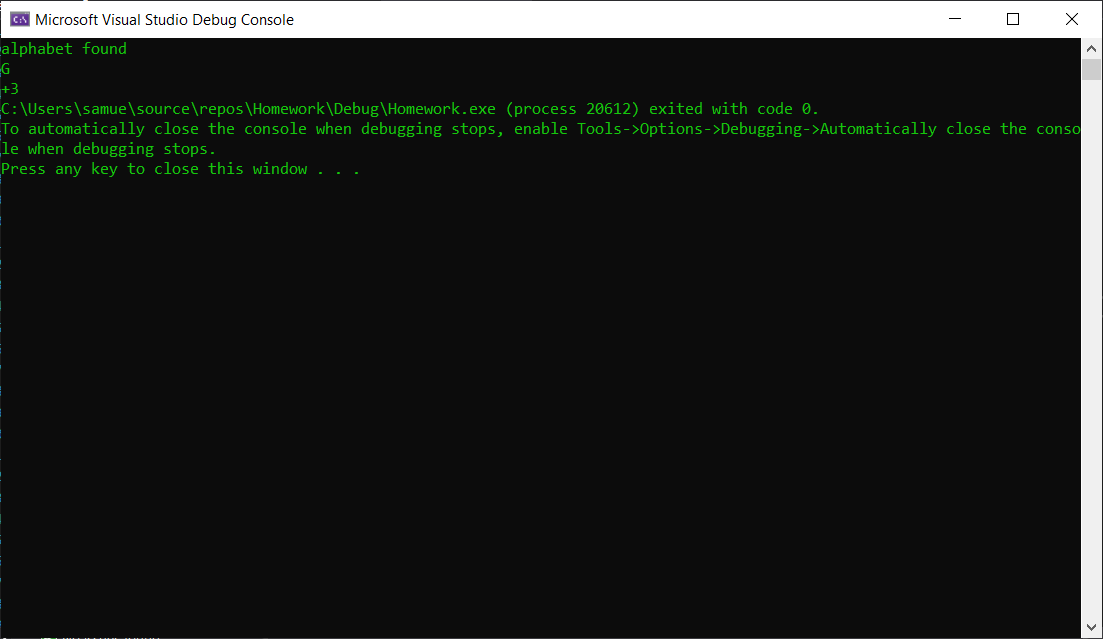
b. Array has only numerals; Boundary Case

c. Several arrays with a mix of alphabets and numerals positioned at different indices

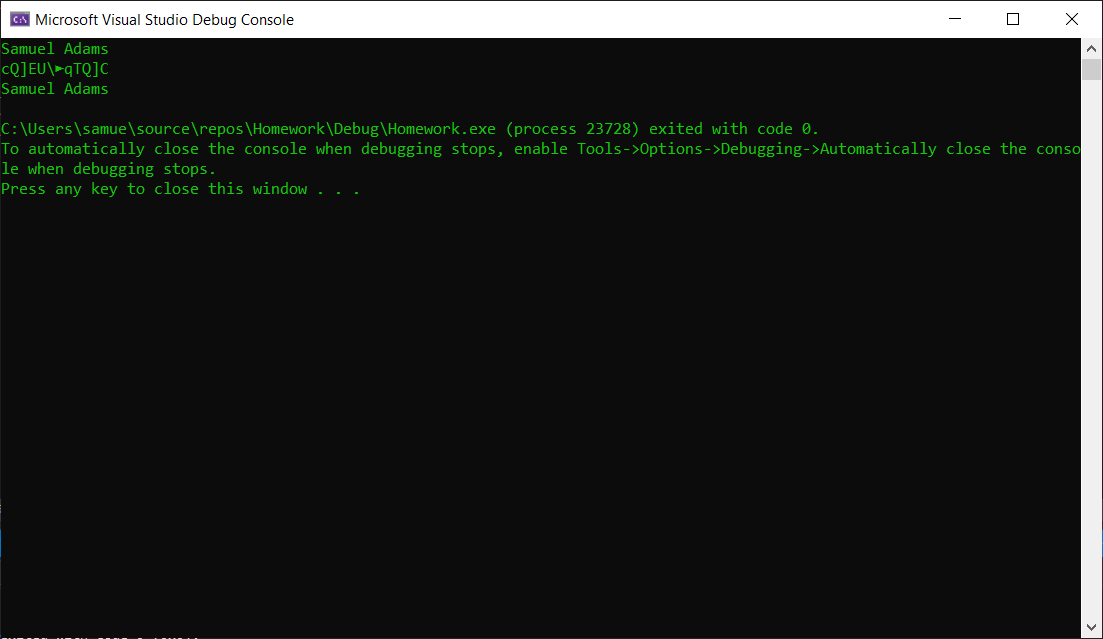
a.



b.

c.

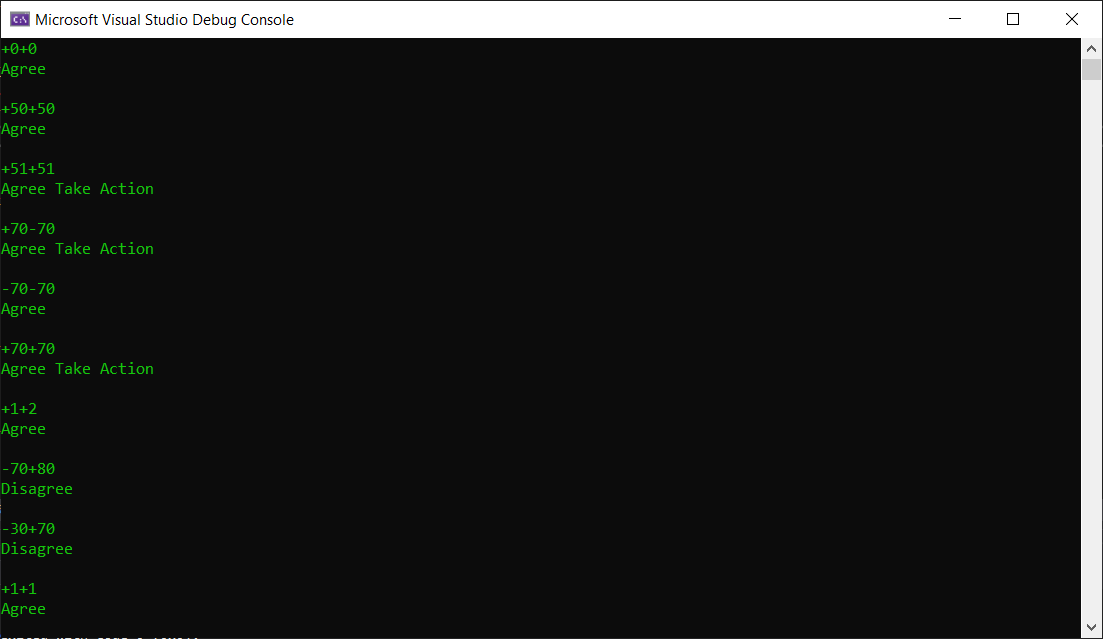
2. Write a program which encodes any string using the XOR instruction.  Test it using your <first name last name> in the data segment to produce cipher text and then decode using the program to get plain text.  Use the last two digits of your student id as the key.  Print plane text from the data segment, 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**)?

The strength is that it is very simple and fast to encrypt using this method as it only requires one key. However, that is also its downfall, the use of only one short key would make it extremely easy to find out this key for the cipher. My name has 3 a’s therefore making it quite easy to see the hexadecimal cipher it uses.

3. Write a program that gets its input from two sensors.  If the values of the sensors differ by no more than +/- 4, print “Agree”, otherwise, print “Disagree.”  You can assume that the values are integers.  Additionally, if the values Agree and they are each more than 50, print “Take Action”. **Submit asm/list file and show screenshots of robust testing for various inputs, including boundary conditions, in the closed interval (-70 … 70).**

****

4. Draw the stack (use, 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 FADD  
   4040026          mov eax, ebx  
            …  
            …                      
  
    Main EndP  
  
 FADD  PROC  
  4041040          Push ecx  
  4041044          Push ebx  
   4041048          mov eax, edx  
            …  
            …  
           404A060         Pop ebx  
           404A062         Pop ecx  
           404A064         ret  
          FADD EndP

|  |  |  |  |
| --- | --- | --- | --- |
| 404001C | 0000 1000 | n/a |  |
|  | 0000 0FFC |  |  |
|  | 0000 0FF8 |  |  |
|  | 0000 0FF4 |  |  |
|  | 0000 0FF0 |  |  |
|  |  |  |  |
| 4041020 | 0000 1000 | n/a |  |
|  | 0000 0FFC | 4041036 |  |
|  | 0000 0FF8 |  |  |
|  | 0000 0FF4 |  |  |
|  | 0000 0FF0 |  |  |
|  |  |  |  |
| 4041040 | 0000 1000 | n/a |  |
|  | 0000 0FFC | 4041036 |  |
|  | 0000 0FF8 |  |  |
|  | 0000 0FF4 |  |  |
|  | 0000 0FF0 |  |  |
| 404A044 | 0000 1000 | n/a |  |
|  | 0000 0FFC | 4041036 |  |
|  | 0000 0FF8 | 0000000Ch | ECX |
|  | 0000 0FF4 |  |  |
|  | 0000 0FF0 |  |  |
| 404A048 | 0000 1000 | n/a |  |
|  | 0000 0FFC | 4041036 |  |
|  | 0000 0FF8 | 0000000Ch | ECX |
|  | 0000 0FF4 | 0000000Bh | EBX |
|  | 0000 0FF0 |  |  |
| 404A060 | 0000 1000 | n/a |  |
|  | 0000 0FFC | 4041036 |  |
|  | 0000 0FF8 | 0000000Ch | ECX |
|  | 0000 0FF4 |  |  |
|  | 0000 0FF0 |  |  |
| 404A062 | 0000 1000 | n/a |  |
|  | 0000 0FFC | 4041036 |  |
|  | 0000 0FF8 |  |  |
|  | 0000 0FF4 |  |  |
|  | 0000 0FF0 |  |  |
| 404A064 | 0000 1000 | n/a |  |
|  | 0000 0FFC |  |  |
|  | 0000 0FF8 |  |  |
|  | 0000 0FF4 |  |  |
|  | 0000 0FF0 |  |  |
| 4040026 | 0000 1000 | n/a |  |
|  | 0000 0FFC |  |  |
|  | 0000 0FF8 |  |  |
|  | 0000 0FF4 |  |  |
|  | 0000 0FF0 |  |  |