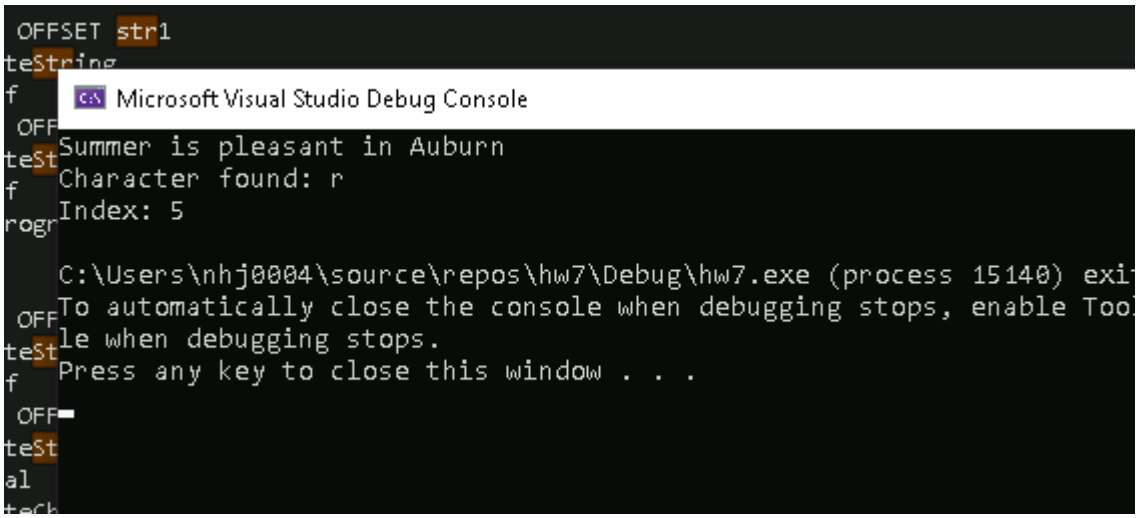
Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

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 a string to determine whether a character in the register AL is present within the string. If found, the program should print “character found”, its value and its index within the string. **Submit the asm/list file and screenshots that show the output of your code for the following example string:**

MyString BYTE "Summer is pleasant in Auburn" 0x0




```
.386

.model flat, stdcall

.stack 4096

ExitProcess proto, dwExitCode:dword


INCLUDE Irvine32.inc


.data

        str1 BYTE "Summer is pleasant in Auburn", 0

        msg1 BYTE "Character found: ", 0

        msg2 BYTE "Index: ", 0
```

Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

```
.code
main PROC
        mov esi, OFFSET str1
        mov al, 'r'
        mov ecx, 0

L1:
        cmp al, [esi]
        je cFound
        inc esi
        inc ecx
        cmp byte ptr [esi], 0
        jne L1

        mov edx, OFFSET str1
        call WriteString
        call Crlf
        mov edx, OFFSET msg1
        call WriteString
        call Crlf
        jmp EndProgram

cFound:
        mov edx, OFFSET str1
        call WriteString
        call Crlf
        mov edx, OFFSET msg1
```

Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

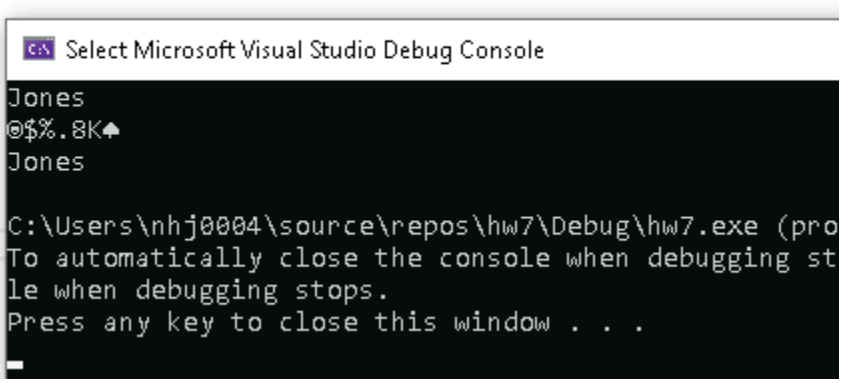
```
        call WriteString

        mov dl, al

        call WriteChar

        call Crlf

        mov edx, OFFSET msg2

        call WriteString

        mov eax, ecx

        call WriteDec

        call Crlf


EndProgram:

        invoke ExitProcess, 0

main ENDP


END main
```

Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

2. Write a program which encodes any string using the XOR instruction. Test it using your <last name> in the data segment to produce ciphertext and then decode using the program to get plain text. Use the last two or three 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 show the output of your code.**

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

- A few of the strengths of using the XOR encryption method are speed and simplicity. XOR encryption is a lot more straightforward and presumably easier to learn in comparison to some other methods of encryption. The speed / efficiency is also very efficient for both encrypting and decrypting. A potential weakness is that undefined length of key. Using a shorter key can make it easier to decrypt. Also, patterns can be seen using the XOR method, and a longer encrypted text is vulnerable to being decoded by hackers by studying the pattern of the encryption.




```
.386
.model flat, stdcall
.stack 4096
ExitProcess proto, dwExitCode:dword

INCLUDE Irvine32.inc

.data
        LastName BYTE "Jones",0
        bufSize DWORD $ - OFFSET LastName
```

Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

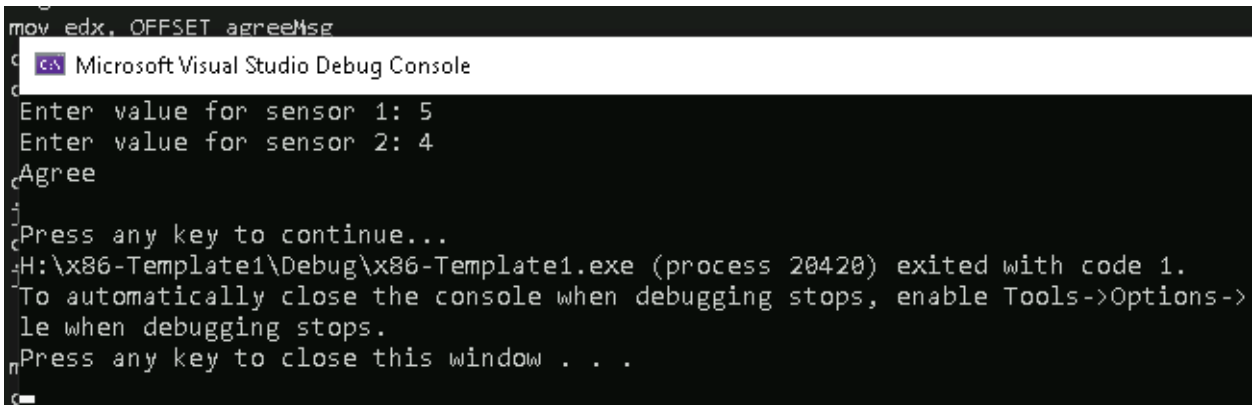
```
        KEY DWORD 59

.code
main PROC
        mov esi, OFFSET LastName
        mov ecx, bufSize
        mov eax, KEY

                ; otuput plaintext
        mov edx, OFFSET LastName
        call WriteString
        call Crlf


        ; encode
        L1:
        xor byte ptr [esi], al
        inc esi
        loop L1


        ; output ciphertext
        mov edx, OFFSET LastName
        call WriteString
        call Crlf

        mov esi, OFFSET lastName
        mov ecx, bufSize
        mov eax, KEY
```

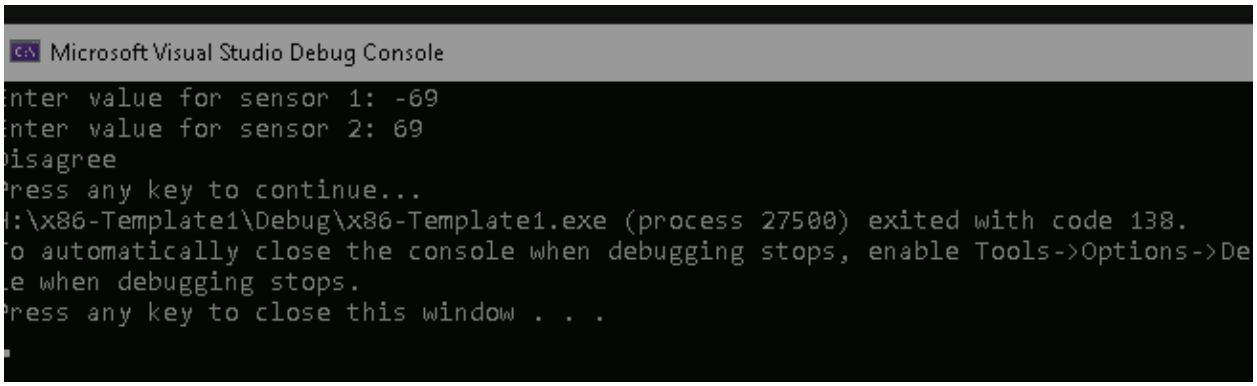
Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

```
        ; decode
        L2:
        xor byte ptr [esi], al
        inc esi
        loop L2

        ; output decrypted text
        mov edx, OFFSET LastName
        call WriteString
        call Crlf

        invoke ExitProcess, 0
main ENDP

END main
```

Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

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 “Nose Down”. **Submit asm/list file and show screenshots of robust testing for various inputs, including boundary conditions, in the closed interval (-70 … 70).**





Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

```
Microsoft Visual Studio Debug Console
Enter value for sensor 1: -69
Enter value for sensor 2: -68
Agree

Press any key to continue...
H:\x86-Template1\Debug\x86-Template1.exe (process 344) exited with code 1.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the
le when debugging stops.
Press any key to close this window . . .
```

```
Microsoft Visual Studio Debug Console
Enter value for sensor 1: 51
Enter value for sensor 2: 51
Agree
Nose Down
Press any key to continue...
H:\x86-Template1\Debug\x86-Template1.exe (process 21164) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging
le when debugging stops.
Press any key to close this window . . .
```

```
Microsoft Visual Studio Debug Console
Enter value for sensor 1: 55
Enter value for sensor 2: 65
Disagree
Press any key to continue...
H:\x86-Template1\Debug\x86-Template1.exe (process 23240) exited with co
To automatically close the console when debugging stops, enable Tools->
le when debugging stops.
Press any key to close this window . . .
```

Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

```
INCLUDE Irvine32.inc

.DATA

sensor1   DWORD ?

sensor2   DWORD ?

diff      DWORD ?

prompt1   BYTE  "Enter value for sensor 1: ",0

prompt2   BYTE  "Enter value for sensor 2: ",0

agreeMsg  BYTE  "Agree",0

disagreeMsg BYTE "Disagree",0

noseDownMsg BYTE "Nose Down",0


.CODE

main PROC

    call Clrscr


    ; sensor 1

    mov edx, OFFSET prompt1

    call WriteString

    call ReadInt

    mov sensor1, eax


    ; input for sensor 2

    mov edx, OFFSET prompt2

    call WriteString

    call ReadInt

    mov sensor2, eax
```

Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

```
    ; abs value
    mov eax, sensor1
    sub eax, sensor2
    mov diff, eax
    cmp eax, 0
    jge difference
    neg eax
    mov diff, eax

difference:
    ; Check if the difference is <= 4
    cmp diff, 4
    jbe agree
    jmp disagree

agree:
    ; check if both sensors are > 50
    cmp sensor1, 50
    jg checkSensor2
    jmp printAgree

checkSensor2:
    cmp sensor2, 50
    jg printAgree
    jmp disagree

printAgree:
    mov edx, OFFSET agreeMsg
```

Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

```
    call WriteString
    call Crlf

    cmp sensor1, 50
    jle skipNoseDown
    cmp sensor2, 50
    jle skipNoseDown

    mov edx, OFFSET noseDownMsg
    call WriteString

skipNoseDown:
    jmp exitProgram

disagree:
    mov edx, OFFSET disagreeMsg
    call WriteString

exitProgram:
    call Crlf
    call WaitMsg
    ret
main ENDP

END main
```

Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

4. Draw the stack (use any handwriting/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, 0ABCDh
404001C       mov ebx, 1234h
***4040020       call FDIV
***4040026       mov eax, ebx
                 …
                 …

Main EndP

FDIV  PROC
***4041040       Push ebx
***4041044       Push ecx
***4041048       mov eax, edx
                 …
                 …
***404A060       Pop ecx
***404A062       Pop ebx
***404A064       ret
FDIV EndP
```

Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

Before : call FDIV

Address	**Instruction**	**Stack**
4040018	mov ecx, 0ABCDh	
404001C	mov ebx, 1234h	
4040020	call FDIV	n/a

Before : mov eax, ebx

Address	**Instruction**	**Stack**
4040018	mov ecx, 0ABCDh	
404001C	mov ebx, 1234h	
4040020	call FDIV	n/a
4040026	mov eax, ebx	

Before : push ebx inside FDIV

Address	**Instruction**	**Stack**
4040018	mov ecx, 0ABCDh	
404001C	mov ebx, 1234h	
4040020	call FDIV	n/a
4040026	mov eax, ebx	
4041040	push ebx	ebx

Before : push ecx inside FDIV

Address	**Instruction**	**Stack**
4040018	mov ecx, 0ABCDh	
404001C	mov ebx, 1234h	
4040020	call FDIV	n/a
4040026	mov eax, ebx	

Noah Jones
Dr. Baskiyar
COMP-3350
Homework 7

4041040	push ebx	ebx
4041044	push ecx	ebx
		ecx

Before : mov eax, edx inside FDIV

Address	**Instruction**	**Stack**
4040018	mov ecx, 0ABCDh	
404001C	mov ebx, 1234h	
4040020	call FDIV	n/a
4040026	mov eax, ebx	
4041040	push ebx	ebx
4041044	push ecx	ebx
4041048	mov eax, edx	ebx
		ecx
		eax

Before : pop ecx,

Address	**Instruction**	**Stack**
4040018	mov ecx, 0ABCDh	
404001C	mov ebx, 1234h	
4040020	call FDIV	n/a
4040026	mov eax	