**Andrew Won**

**6.11.2 Programming Exercises**

5.

INCLUDE Irvine32.inc

.data

msgMenu BYTE "---- Boolean Calculator ----", 0dh, 0ah

BYTE 0dh,0ah

BYTE "1. x AND y" , 0dh, 0ah

BYTE "2. x OR y" , 0dh, 0ah

BYTE "3. NOT x" , 0dh, 0ah

BYTE "4. x XOR y" , 0dh, 0ah

BYTE "5. Exit program", 0

msgAND BYTE "Boolean AND", 0

msgOR BYTE "Boolean OR", 0

msgNOT BYTE "Boolean NOT", 0

msgXOR BYTE "Boolean XOR", 0

caseTable BYTE '1'

DWORD AND\_op

EntrySize = ($ - caseTable )

BYTE '2'

DWORD OR\_op

BYTE '3'

DWORD NOT\_op

BYTE '4'

DWORD XOR\_op

BYTE '5'

DWORD ExitProgram

NumberOfEntries = ($ - caseTable) / EntrySize

.code

main PROC

call Clrscr

Menu:

Mov edx, OFFSET msgMenu

Call WriteString

Call Crlf

L1:

Call ReadChar

Cmp al,'5'

Ja L1

Cmp al,'1'

jb L1

call ChooseProcedure

jc quit

call Crlf

call WriteString

call Crlf

call Crlf

jmp Menu

quit: exit

main ENDP

ChooseProcedure PROC

Push ebx

push ecx

mov ebx, OFFSET caseTable

mov ecx, NumberOfEntries

L1:

Cmp al, [ebx]

Jne L2

Call NEAR PTR [ebx + 1]

Jmp L3

L2:

add ebx, EntrySize

loop L1

L3:

pop ecx

pop ebx

ret

ChooseProcedure ENDP

AND\_op PROC

mov edx, OFFSET msgAND

ret

AND\_op ENDP

OR\_op PROC

mov edx, OFFSET msgOR

ret

OR\_op ENDP

NOT\_op PROC

mov edx, OFFSET msgNOT

ret

NOT\_op ENDP

XOR\_op PROC

mov edx, OFFSET msgXOR

ret

XOR\_op ENDP

ExitProgram PROC

stc

ret

ExitProgram ENDP

END main

6.

INCLUDE Irvine32.inc

.data

msgMenu BYTE "---- Boolean Calculator ----------", 0dh, 0ah

BYTE 0dh, 0ah

BYTE "1. x AND y" , 0dh, 0ah

BYTE "2. x OR y" , 0dh, 0ah

BYTE "3. NOT x" , 0dh, 0ah

BYTE "4. x XOR y" , 0dh, 0ah

BYTE "5. Exit program", 0dh, 0ah, 0dh, 0ah

BYTE "Enter integer> ", 0

msgAND BYTE "Boolean AND", 0

msgOR BYTE "Boolean OR", 0

msgNOT BYTE "Boolean NOT", 0

msgXOR BYTE "Boolean XOR", 0

msgOperand1 BYTE "Input the first 32-bit hexadecimal operand: ", 0

msgOperand2 BYTE "Input the second 32-bit hexadecimal operand: ", 0

msgResult BYTE "The 32-bit hexadecimal result is: ", 0

caseTable BYTE '1'

DWORD AND\_op

EntrySize = ($ - caseTable )

BYTE '2'

DWORD OR\_op

BYTE '3'

DWORD NOT\_op

BYTE '4'

DWORD XOR\_op

BYTE '5'

DWORD ExitProgram

NumberOfEntries = ($ - caseTable) / EntrySize

.code

main PROC

call Clrscr

Menu:

Mov edx, OFFSET msgMenu

call WriteString

L1:

call ReadChar

cmp al, '5'

ja L1

cmp al, '1'

jb L1

call Crlf

call ChooseProcedure

jc quit

call Crlf

jmp Menu

quit:

exit

main ENDP

ChooseProcedure PROC

push ebx

push ecx

mov ebx,OFFSET caseTable

mov ecx,NumberOfEntries

L1:

cmp al, [ebx]

jne L2

call NEAR PTR [ebx + 1]

jmp L3

L2:

add ebx, EntrySize

loop L1

L3:

pop ecx

pop ebx

ret

ChooseProcedure ENDP

AND\_op PROC

pushad

mov edx, OFFSET msgAND

call WriteString

call Crlf

call Crlf

mov edx, OFFSET msgOperand1

call WriteString

call ReadHex

mov ebx, eax

mov edx, OFFSET msgOperand2

call WriteString

call ReadHex

and eax, ebx

mov edx, OFFSET msgResult

call WriteString

call WriteHex

call Crlf

popad

ret

AND\_op ENDP

OR\_op PROC

pushad

mov edx, OFFSET msgOR

call WriteString

call Crlf

call Crlf

mov edx, OFFSET msgOperand1

call WriteString

call ReadHex

mov ebx, eax

mov edx, OFFSET msgOperand2

call WriteString

call ReadHex

or eax, ebx

mov edx, OFFSET msgResult

call WriteString

call WriteHex

call Crlf

popad

ret

OR\_op ENDP

NOT\_op PROC

pushad

mov edx, OFFSET msgNOT

call WriteString

call Crlf

call Crlf

mov edx, OFFSET msgOperand1

call WriteString

call ReadHex

not eax

mov edx, OFFSET msgResult

call WriteString

call WriteHex

call Crlf

popad

ret

NOT\_op ENDP

XOR\_op PROC

pushad

mov edx, OFFSET msgXOR

call WriteString

call Crlf

call Crlf

mov edx, OFFSET msgOperand1

call WriteString

call ReadHex

mov ebx, eax

mov edx, OFFSET msgOperand2

call WriteString

call ReadHex

xor eax,ebx

mov edx, OFFSET msgResult

call WriteString

call WriteHex

call Crlf

popad

ret

XOR\_op ENDP

ExitProgram PROC

stc

ret

ExitProgram ENDP

END main