Activity 10

Conditional Jumps Based on Specific Flags

Mnemonic	Description	Flags
JZ	Jump if zero	ZF = 1
JNZ	Jump if not zero	ZF = 0
JC	Jump if carry	CF = 1
JNC	Jump if not carry	CF = 0
JO	Jump if overflow	OF = 1
JNO	Jump if not overflow	OF = 0
JS	Jump if signed	SF = 1
JNS	Jump if not signed	SF = 0
JP	Jump if parity (even)	PF = 1
JNP	Jump if not parity (odd)	PF = 0

Conditional Jumps Based on Equality

Mnemonic	Description
JE	Jump if equal (*leftOp* = *rightOp*)
JNE	Jump if not equal (*leftOp* ≠ *rightOp*)
JCXZ	Jump if CX = 0
JECXZ	Jump if ECX = 0

Conditional Jumps Based on Unsigned Comparisons

Mnemonic	Description
JA	Jump if above (if *leftOp* > *rightOp*)
JNBE	Jump if not below or equal (same as JA)
JAE	Jump if above or equal (if *leftOp* >= *rightOp*)
JNB	Jump if not below (same as JAE)
JB	Jump if below (if *leftOp* < *rightOp*)
JNAE	Jump if not above or equal (same as JB)
JBE	Jump if below or equal (if *leftOp* <= *rightOp*)
JNA	Jump if not above (same as JBE)

Conditional Jumps Based on Signed Comparisons

Mnemonic	Description
JG	Jump if greater (if *leftOp* > *rightOp*)
JNLE	Jump if not less than or equal (same as JG)
JGE	Jump if greater than or equal (if *leftOp* >= *rightOp*)
JNL	Jump if not less (same as JGE)
JL	Jump if less (if *leftOp* < *rightOp*)
JNGE	Jump if not greater than or equal (same as JL)
JLE	Jump if less than or equal (if *leftOp* <= *rightOp*)
JNG	Jump if not greater (same as JLE)

EXERCISES

Pay careful attention to whether you are reading and comparing signed *or* unsigned *integers. Recall:* `ReadInt` *and* `ReadDec` *read signed an unsigned integers, respectively.*

1. (a)

```
        call ReadInt
        mov ebx, eax  ; Store first value in EBX
        call ReadInt
        add eax, ebx  ; Store sum in EAX
        _____?_____
        call WriteInt
done:   exit
```

(b)

```
        call ReadDec
        mov ebx, eax  ; Store first value in EBX
        call ReadDec
        add eax, ebx  ; Store sum in EAX
        ____?____
        call WriteDec
done:   exit
```

Inserting a conditional jump to *done* will skip over the *call* instruction that displays the sum. Consider the following. What conditional jump instruction should you insert?

i. Don't display the sum iff it is out of range (a) _____ (b) _____

ii. Don't display the sum iff it is negative (a) _____ (b) N/A

iii. Don't display the sum iff it is zero (a) _____ (b) _____

2. (a)

```
        call ReadInt
        mov esi, eax  ; Store first value in ESI
        call ReadInt
        cmp esi, eax  ; Compare first & second values
        _____?_____
        jmp done
disp:   call WriteInt
done:   exit
```

(b)

```
        call ReadDec
        mov esi, eax  ; Store first value in ESI
        call ReadDec
        cmp esi, eax  ; Compare first & second values
        ____?____
        jmp done
disp:   call WriteInt
done:   exit
```

Inserting a conditional jump to *disp* will display the second value. Consider the following. What conditional jump instruction should you insert?

i. Display the second value iff the first value < second value (a) _____ (b) _____

ii. Display the second value iff the first value ≤ second value (a) _____ (b) _____

iii. Display the second value iff the first value > second value (a) _____ (b) _____

iv. Display the second value iff the first value ≥ second value (a) _____ (b) _____

v. Display the second value iff the first value == second value (a) _____ (b) _____

vi. Display the second value iff the first value ≠ second value (a) _____ (b) _____