National University of Computer and Emerging Sciences



**Laboratory Manual**

*for*

**Computer Organization and Assembly Language Programming**

| Course Instructor | Aleena Ahmad |
| --- | --- |
| Lab Instructor | Sana Ejaz |
| Semester | Fall 2024 |

Department of Computer Science

FAST-NU, Lahore, Pakistan

**OBJECTIVES:**

∙ How to implement Conditional jumps.

∙ How to implement Unconditional jumps.

∙ How to implement sorting using jumps.

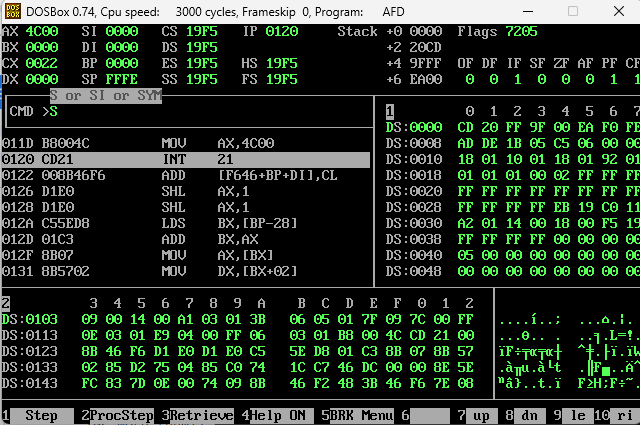
**Instructions:**

| ∙ Run and debug the programs, ensuring that they behave as expected.  ∙ Submit work in a single Word file with Code and screenshots. No asm, lst , or com. (Do not submit a zip folder).  ∙ Document your observations and note any issues encountered during implementation in the same word document as the code and screenshots. |
| --- |

**Task 1: Write an assembly language program that compares two numbers stored in memory. If the first number is greater than the second, jump to a label that increments the first number by 1. Otherwise, jump to a label that decrements the first number by 1.**

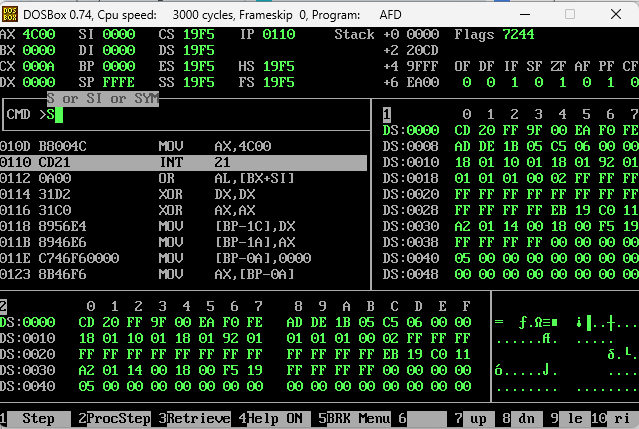
*Hint: Use CMP, JG (jump if greater), and JMP instructions.*

| [org 0x0100]  JMP start  num1: dw 10  num2: dw 20  start:  MOV AX, [num1]  CMP AX, [num2]  JG greater  JL lesser  lesser:  DEC word [num1]  JMP terminate  greater:  INC word [num1]  terminate:  MOV AX, 0x4C00  INT 0x21 |
| --- |

**

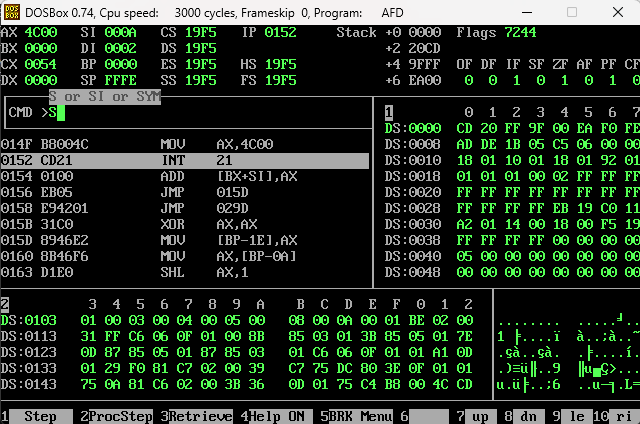
**Task 2: Write a program that continuously loops through a series of instructions until a specific condition is met. For this task, increment a counter until it reaches a certain value, and then use an unconditional jump to exit the loop.** *Hint: Use JMP for the unconditional jump and CMP for condition checking.*

| [org 0x0100]  MOV CX, 0  lop:  INC CX  CMP CX, [num]  JNE lop    JMP terminate  terminate:  MOV AX, 0x4C00  INT 0x21  num: dw 10 |
| --- |

**

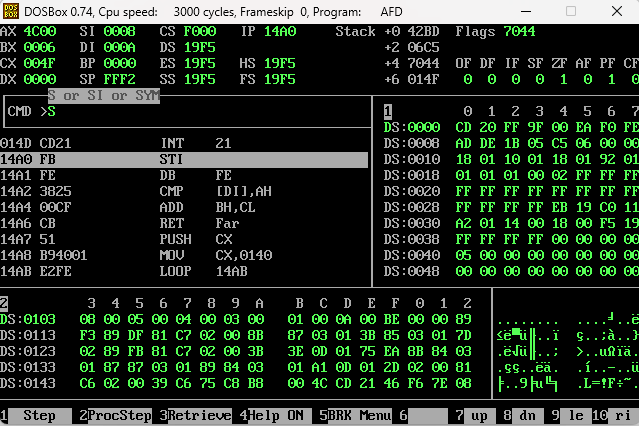
**Task 3: Implement a bubble sort algorithm to sort an array of 5 numbers in ascending order using conditional and unconditional jumps.**

**arr dw 5, 3, 8, 4, 1.**

****

| [org 0x0100]  JMP start  arr: dw 5, 3, 8, 4, 1  size: dw size - arr  swap: db 0  start:  MOV SI, 2    outerloop:  XOR DI, DI  MOV byte [swap], 0  innerloop:  MOV AX, [arr + DI]  CMP AX, [arr + DI + 2]  JLE noswap  XCHG AX, [arr + DI + 2]  XCHG AX, [arr + DI]  MOV byte [swap], 1  noswap:  MOV AX, [size]  SUB AX, SI  ADD DI, 2  CMP DI, AX  JNE innerloop  CMP byte [swap], 1  JNE terminate  ADD SI, 2  CMP SI, [size]  JNE outerloop  terminate:  MOV AX, 0x4C00  INT 0x21 |
| --- |

**Task 4: Implement a selection sort algorithm to sort an array of 6 numbers in descending order. Use jumps for the comparison and swapping logic.** *Hint: Use two loops.*

**

| [org 0x0100]  JMP start  arr: dw 5, 3, 8, 4, 1  size: dw size - arr  start:  MOV SI, 0    outerloop:  MOV BX, SI  MOV DI, BX  ADD DI, 2  innerloop:  MOV AX, [arr + BX]  CMP AX, [arr + DI]  JNL nobig    MOV BX, DI  nobig:  ADD DI, 2  CMP DI, [size]  JNE innerloop  noswap:  MOV AX, [arr + SI]  XCHG AX, [arr + BX]  MOV [arr + SI], AX  MOV AX, [size]  SUB AX, 2  ADD SI, 2  CMP SI, AX  JNE outerloop  terminate:  MOV AX, 0x4C00  INT 0x21 |
| --- |