```
.MODEL SMALL
.STACK 100H

.DATA
N DB ?                      ; number of students input by user
student_numbers DB 10 DUP (?)   ; max 10 students for demo
student_grades DB 10 DUP (?)

MSG_N DB 'Enter number of students (max 10): $'
MSG_SN DB 'Enter student number (single digit): $'
MSG_GR DB 'Enter student grade (single digit): $'
MSG1 DB 'Student Numbers: $'
MSG2 DB 'Grades: $'
NEWLINE DB 13, 10, '$'          ; CR LF

.CODE
MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

    ; ===== Print newline, then Input N =====
    MOV DX, OFFSET NEWLINE
    MOV AH, 09H
    INT 21H

    MOV DX, OFFSET MSG_N
    MOV AH, 09H
    INT 21H

    MOV AH, 01H            ; input char function
    INT 21H
    SUB AL, 30H            ; convert ASCII to number
    MOV N, AL

    ; ===== Input student numbers =====
    MOV CL, N
    MOV CH, 0
    MOV SI, 0              ; index

INPUT_SN_LOOP:
    ; Print newline before each prompt
    MOV DX, OFFSET NEWLINE
    MOV AH, 09H
    INT 21H

    MOV DX, OFFSET MSG_SN
    MOV AH, 09H
    INT 21H
```

```
    MOV AH, 01H
    INT 21H
    SUB AL, 30H          ; convert ASCII digit to number
    MOV BYTE PTR [student_numbers + SI], AL

    INC SI
    LOOP INPUT_SN_LOOP

    ; ===== Input student grades =====
    MOV CL, N
    MOV CH, 0
    MOV SI, 0

INPUT_GR_LOOP:
    ; Print newline before each prompt
    MOV DX, OFFSET NEWLINE
    MOV AH, 09H
    INT 21H

    MOV DX, OFFSET MSG_GR
    MOV AH, 09H
    INT 21H

    MOV AH, 01H
    INT 21H
    SUB AL, 30H          ; convert ASCII digit to number
    MOV BYTE PTR [student_grades + SI], AL

    INC SI
    LOOP INPUT_GR_LOOP

    ; ========== Sorting (Bubble Sort Descending) ==========
    MOV CL, N
    MOV CH, 0
    DEC CX               ; CX = N-1 (number of passes)

SORT_OUTER_LOOP:
    MOV CH, CL           ; CH = current pass count
    MOV SI, 0            ; SI = index

SORT_INNER_LOOP:
    MOV AL, BYTE PTR [student_grades + SI]
    MOV AH, BYTE PTR [student_grades + SI + 1]
    CMP AL, AH
    JAE NO_SWAP

    ; Swap grades
```

```
    XCHG AL, AH
    MOV BYTE PTR [student_grades + SI], AL
    MOV BYTE PTR [student_grades + SI + 1], AH

    ; Swap student numbers accordingly
    MOV AL, BYTE PTR [student_numbers + SI]
    MOV AH, BYTE PTR [student_numbers + SI + 1]
    XCHG AL, AH
    MOV BYTE PTR [student_numbers + SI], AL
    MOV BYTE PTR [student_numbers + SI + 1], AH

NO_SWAP:
    INC SI
    DEC CH
    JNZ SORT_INNER_LOOP

    DEC CL
    JNZ SORT_OUTER_LOOP

    ; ========== Display Student Numbers ==========
    MOV DX, OFFSET MSG1
    MOV AH, 09H
    INT 21H

    MOV SI, 0
    MOV CL, N
    MOV CH, 0

DISPLAY_SN_LOOP:
    MOV AL, BYTE PTR [student_numbers + SI]
    ADD AL, 30H
    MOV DL, AL
    MOV AH, 02H
    INT 21H

    ; Space
    MOV DL, 20H
    MOV AH, 02H
    INT 21H

    INC SI
    DEC CL
    JNZ DISPLAY_SN_LOOP

    ; Newline
    MOV DX, OFFSET NEWLINE
    MOV AH, 09H
    INT 21H
```

```
    ; ========== Display Grades ==========
    MOV DX, OFFSET MSG2
    MOV AH, 09H
    INT 21H

    MOV SI, 0
    MOV CL, N
    MOV CH, 0

DISPLAY_GR_LOOP:
    MOV AL, BYTE PTR [student_grades + SI]
    ADD AL, 30H
    MOV DL, AL
    MOV AH, 02H
    INT 21H

    ; Space
    MOV DL, 20H
    MOV AH, 02H
    INT 21H

    INC SI
    DEC CL
    JNZ DISPLAY_GR_LOOP

    ; Newline
    MOV DX, OFFSET NEWLINE
    MOV AH, 09H
    INT 21H

    ; ========== Exit ==========
    MOV AH, 4CH
    INT 21H

MAIN ENDP
END MAIN
```
