AHSANULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY

Department of Computer Science and Engineering
Program: BSc in Computer Science and Engineering

Course Code: CSE 2214

Assignment No: 04

Date of Submission: 16/08/2025

Submitted by,
Name: Aaheed Bin Ashraf
Student ID: 20230104094
Section: B2
Year : 2.2

1. **Write a program that lets the user type some text, consisting of words separated by blanks, ending with a carriage return, and displays the text in the same word order as entered, but with the letters in each word reversed.**

ANSWER:

```
.MODEL SMALL
.STACK 100H

.DATA
    INPUT_MAX  EQU 100



    INPUT   DB  INPUT_MAX
            DB  0
            DB  INPUT_MAX DUP(?)

    REVERSE DB  INPUT_MAX+1 DUP('$')

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

    LEA DX, INPUT
    MOV AH, 0Ah
    INT 21h

    LEA SI, INPUT+2
    MOV CL, [INPUT+1]
    XOR CH, CH
    LEA DI, REVERSE
```

```
NEXT_CHUNK:
    CMP CX, 0
    JE  FINISH

    MOV AL, [SI]
    CMP AL, ' '
    JNE WORD_START

SPACE_COPY:
    MOV [DI], AL
    INC DI
    INC SI
    DEC CX
    JZ  FINISH
    MOV AL, [SI]
    CMP AL, ' '
    JE  SPACE_COPY
    JMP NEXT_CHUNK

WORD_START:
    MOV BX, SI
    MOV DX, CX
FIND_END:
    CMP DX, 0
    JE  WORD_END_POS
    MOV AL, [SI]
    CMP AL, ' '
    JE  WORD_END_POS
    INC SI
    DEC DX
    JMP FIND_END
```

```
WORD_END_POS:

    MOV BP, SI
    MOV CX, DX
    DEC SI

REV_COPY:
    MOV AL, [SI]
    MOV [DI], AL
    INC DI
    CMP SI, BX
    JE  WORD_DONE
    DEC SI
    JMP REV_COPY

WORD_DONE:
    MOV SI, BP
    JMP NEXT_CHUNK

FINISH:
    MOV BYTE PTR [DI], '$'
    MOV AH, 2
    MOV DL, 0Dh
    INT 21h
    MOV DL, 0Ah
    INT 21h

    LEA DX, REVERSE
    MOV AH, 9
    INT 21h
    MOV AH, 4Ch
    INT 21h

MAIN ENDP
END MAIN
```

2. Write a program that lets the user type in an algebraic expression, ending with a carriage return, that contains round (parentheses), square, and curly brackets. As the expression is being typed in, the program evaluates each character. If at any point the expression is incorrectly bracketed (too many right brackets or a mismatch between left and right brackets), the program tells the user to start over. After the carriage return is typed, if the expression is correct, the program displays "expression is correct." If not, the program displays "too many left brackets". In both cases, the program asks the user if he or she wants to continue. If the user types 'Y', the program runs again. Your program does not need to store the input string, only check it for correctness.

ANSWER:

```
.DATA
    FIRST       DB 'Enter Expression: $'
    VALID       DB 'Expression is correct.$'
    CONTINUE    DB 13, 10, 'Do you want to continue (Y/N)? $'
    L_WRONG     DB 'Too many left brackets.$'
    R_WRONG     DB 'Mismatch or too many right brackets.$'
    NOT_MATCH   DB 'Bracket mismatch.$'

    STACK       DW 100 DUP(0)
    TOP         DW 0

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

```
BEGIN:
    LEA DX, FIRST
    MOV AH, 9
    INT 21H

    XOR CX, CX
    MOV AH, 1

INPUT:
    INT 21H
    CMP AL, 0DH
    JE ENTER_PRESSED

    CMP AL, '('
    JE PUSH_BRACKET
    CMP AL, '{'
    JE PUSH_BRACKET
    CMP AL, '['
    JE PUSH_BRACKET

    CMP AL, ')'
    JE FIRST_BRACKET
    CMP AL, '}'
    JE SECOND_BRACKET
    CMP AL, ']'
    JE THIRD_BRACKET

    JMP INPUT

PUSH_BRACKET:
    PUSH AX
    INC CX
    JMP INPUT

FIRST_BRACKET:
    POP DX
    DEC CX
    CMP CX, 0
    JL RIGHT_BRACKET_ERROR
    CMP DL, '('
    JNE NO_MATCH
    JMP INPUT
```

```
SECOND_BRACKET:
    POP DX
    DEC CX
    CMP CX, 0
    JL RIGHT_BRACKET_ERROR
    CMP DL, '{'
    JNE NO_MATCH
    JMP INPUT

THIRD_BRACKET:
    POP DX
    DEC CX
    CMP CX, 0
    JL RIGHT_BRACKET_ERROR
    CMP DL, '['
    JNE NO_MATCH
    JMP INPUT

ENTER_PRESSED:
    CMP CX, 0
    JNE LEFT_BRACKETS_ERROR

    MOV AH, 9
    LEA DX, VALID
    INT 21H
    LEA DX, CONTINUE
    INT 21H

    MOV AH, 1
    INT 21H
    CMP AL, 'Y'
    JNE EXIT
    JMP BEGIN

NO_MATCH:
    LEA DX, NOT_MATCH
    MOV AH, 9
    INT 21H
    JMP BEGIN

LEFT_BRACKETS_ERROR:
    LEA DX, L_WRONG
    MOV AH, 9
    INT 21H
    JMP BEGIN

RIGHT_BRACKET_ERROR:
    LEA DX, R_WRONG
    MOV AH, 9
    INT 21H
    JMP BEGIN

EXIT:
    MOV AH, 4CH
    INT 21H
MAIN ENDP
END MAIN
```