Ahsanullah University of Science & Technology

Department of Computer Science and Engineering

Course No : CSE 2214

Course Title : Assembly Language Programming Sessional

Assignment no : 08

Date of Performance : 19.08.20

Date of Submission : 25.08.20

Submitted To : Ms.Tahsin Aziz & Md.Siam Ansary

Submitted By:

Name : Mahin opu

ID : 17.02.04.006

Year : Second

Semester : 2nd

Group : A1

Section : A

Question 01:

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
   PROMPT_1  DB  'Enter the string : $'
   PROMPT_2  DB  0DH,0AH,'The string with words in reverse order : $'
   COUNT     DW  0
 .CODE
   MAIN PROC
     MOV AX, @DATA
     MOV DS, AX
     LEA DX, PROMPT_1
     MOV AH, 9
     INT 21H
     XOR CX, CX
     MOV AH, 1
     @INPUT:
      INT 21H
      CMP AL, 0DH
      JE @END_INPUT
```

```
  PUSH AX
  INC CX
JMP @INPUT
@END_INPUT:
MOV BX, 50H
XCHG BX, SP
PUSH 0020H
XCHG BX, SP
INC COUNT
@LOOP_1:
  POP DX
  XCHG BX, SP
  PUSH DX
  XCHG BX, SP
  INC COUNT
LOOP @LOOP_1
LEA DX, PROMPT_2
MOV AH, 9
INT 21H
MOV CX, COUNT
MOV COUNT, 0
PUSH 0020H
INC COUNT
```

```
 @OUTPUT:
   XCHG BX, SP
   POP DX
   XCHG BX, SP
   CMP DL, 20H
   JNE @SKIP_PRINTING
   MOV AH, 2
  @LOOP_2:
   POP DX
   INT 21H
 DEC COUNT
 JNZ @LOOP_2
 MOV DX, 0020H
 @SKIP_PRINTING:
 PUSH DX
 INC COUNT
LOOP @OUTPUT
MOV AH, 4CH
INT 21H
MAIN ENDP
END MAIN
```

Question 02:

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:

```
.MODEL SMALL

 .STACK 100H

 .DATA

  PROMPT         DB  0DH,0AH,'Enter an Algebraic Expression : ',0DH,0AH,'$'

  CORRECT        DB  0DH,0AH,'Expression is Correct.$'

  LEFT_BRACKETS   DB  0DH,0AH,'Too many Left Brackets.$'

  RIGHT_BRACKETS  DB  0DH,0AH,'Too many Right Brackets.$'

  MISMATCH       DB  0DH,0AH,'Bracket Mismatch. Begin Again.$'

  CONTINUE       DB  0DH,0AH,'Type Y if you want to Continue : $'


 .CODE

  MAIN PROC

   MOV AX, @DATA
MOV DS, AX
 @START:
 LEA DX, PROMPT
 MOV AH, 9
 INT 21H
 XOR CX, CX
 MOV AH, 1
 @INPUT:
  INT 21H
  CMP AL, 0DH
  JE @END_INPUT
  CMP AL, "["
  JE @PUSH_BRACKET
  CMP AL, "{"
  JE @PUSH_BRACKET
  CMP AL, "("
  JE @PUSH_BRACKET
  CMP AL, ")"
  JE @ROUND_BRACKET
  CMP AL, "}"
  JE @CURLY_BRACKET
  CMP AL, "]"
  JE @SQUARE_BRACKET
```

```
JMP @INPUT

@PUSH_BRACKET:

PUSH AX

INC CX

JMP @INPUT

@ROUND_BRACKET:

POP DX

DEC CX

CMP CX, 0

JL @RIGHT_BRACKETS

CMP DL, "("

JNE @MISMATCH

JMP @INPUT

@CURLY_BRACKET:

POP DX

DEC CX

CMP CX, 0

JL @RIGHT_BRACKETS

CMP DL, "{"

JNE @MISMATCH

JMP @INPUT

@SQUARE_BRACKET:

POP DX

DEC CX
```

```
  CMP CX, 0
 JL @RIGHT_BRACKETS
  CMP DL, "["
  JNE @MISMATCH
JMP @INPUT
@END_INPUT:
CMP CX, 0
JNE @LEFT_BRACKETS
MOV AH, 9
LEA DX, CORRECT
INT 21H
LEA DX, CONTINUE
INT 21H
MOV AH, 1
INT 21H
CMP AL, "Y"
JNE @EXIT
JMP @START
@MISMATCH:
LEA DX, MISMATCH
MOV AH, 9
INT 21H
JMP @START
@LEFT_BRACKETS:
 LEA DX, LEFT_BRACKETS
 MOV AH, 9
 INT 21H
  JMP @START
 @RIGHT_BRACKETS:
 LEA DX, RIGHT_BRACKETS
 MOV AH, 9
 INT 21H
 JMP @START
 @EXIT:
MOV AH, 4CH
INT 21H
MAIN ENDP
END MAIN
```