Ahsanullah University of Science & Technology

Department of Computer Science and Engineering

Course No : CSE 2214

Course Title : Assembly Language Programming Sessional

Assignment no : 09

Date of Performance : 26.08.20

Date of Submission : 01.09.20

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

Submitted By:

Name : Mahin opu

ID : 17.02.04.006

Year : 2nd

Semester : 2nd

Group : A1

Section : A

Question 1:

Write a program that lets the user enter time in seconds, up to 65535 and outputs the time as hours, minutes, and seconds.

Answer:

```
.MODEL SMALL
 .STACK 100H

 .DATA
  PROMPT_1  DB  'Enter the time in seconds up to 65535 : $'
  PROMPT_2  DB  0DH,0AH,'The time in HH:MM:SS format is : $'
  SEPARATOR DB  ' : $'

 .CODE
     MAIN PROC
     MOV AX, @DATA
     MOV DS, AX
     LEA DX, PROMPT_1
     MOV AH, 9
     INT 21H
     CALL INDEC
     PUSH AX
```

```
LEA DX, PROMPT_2

MOV AH, 9

INT 21H

POP AX

XOR DX,DX

MOV CX,3600

DIV CX

CMP AX, 10

JGE @HOURS

PUSH AX

MOV AX, 0

CALL OUTDEC

POP AX

@HOURS:

CALL OUTDEC

MOV AX,DX

PUSH AX

LEA DX, SEPARATOR

MOV AH, 9

INT 21H

POP AX

XOR DX,DX
```

```
MOV CX, 60

DIV CX

CMP AX, 10

JGE @MINUTES

PUSH AX

MOV AX, 0

CALL OUTDEC

POP AX

@MINUTES:

CALL OUTDEC

MOV BX,DX

LEA DX, SEPARATOR

MOV AH, 9

INT 21H

MOV AX, BX

CMP AX, 10

JGE @SECONDS

PUSH AX

MOV AX, 0

CALL OUTDEC

POP AX

@SECONDS:
 CALL OUTDEC
 MOV AH, 4CH
 INT 21H
MAIN ENDP
INDEC PROC
PUSH BX
PUSH CX
PUSH DX
JMP @READ
@SKIP_BACKSPACE:
MOV AH, 2
MOV DL, 20H
INT 21H
@READ:
XOR BX,BX
XOR CX,CX
XOR DX,DX
MOV AH, 1
INT 21H
CMP AL, "-"
JE @MINUS
CMP AL, "+"
```

```
JE @PLUS
JMP @SKIP_INPUT
@MINUS:
MOV CH, 1
INC CL
JMP @INPUT
@PLUS:
MOV CH, 2
INC CL
@INPUT:
 MOV AH, 1
 INT 21H
 @SKIP_INPUT:
 CMP AL, 0DH
 JE @JUMP_TO_END_INPUT
 CMP AL, 8H
 JNE @NOT_BACKSPACE
 CMP CH, 0
 JNE @CHECK_REMOVE_MINUS
CMP CL, 0
JE @SKIP_BACKSPACE
JMP @MOVE_BACK
```

```
  @JUMP_TO_END_INPUT:
   JMP @END_INPUT
  @CHECK_REMOVE_MINUS:
  CMP CH, 1
  JNE @CHECK_REMOVE_PLUS
 CMP CL, 1
 JE @REMOVE_PLUS_MINUS
@CHECK_REMOVE_PLUS:
 CMP CL, 1
JE @REMOVE_PLUS_MINUS
JMP @MOVE_BACK
 @REMOVE_PLUS_MINUS:
  MOV AH, 2
  MOV DL, 20H
  INT 21H
 MOV DL, 8H
 INT 21H
 JMP @READ
@MOVE_BACK:
MOV AX, BX
MOV BX, 10
DIV BX
```

```
MOV BX, AX
MOV AH, 2
MOV DL, 20H
INT 21H
MOV DL, 8H
 INT 21H

XOR DX, DX
DEC CL
JMP @INPUT
@NOT_BACKSPACE:
 INC CL
 CMP AL, 30H
 JL @ERROR
 CMP AL, 39H
  JG @ERROR
  AND AX, 000FH
 PUSH AX
 MOV AX, 10
  MUL BX
  MOV BX, AX
 POP AX
 ADD BX, AX
 JC @ERROR
 CMP CL, 5
 JG @ERROR
 JMP @INPUT
@ERROR:
 MOV AH, 2
 MOV DL, 7H
 INT 21H
XOR CH, CH
@CLEAR:
 MOV DL, 8H
INT 21H
MOV DL, 20H
INT 21H
MOV DL, 8H
INT 21H
LOOP @CLEAR
JMP @READ
@END_INPUT:
CMP CH, 1
JNE @EXIT
```

```
NEG BX
@EXIT:
MOV AX, BX
 POP DX
 POP CX
 POP BX
 RET
 INDEC ENDP
 OUTDEC PROC
   PUSH BX
   PUSH CX
   PUSH DX
   CMP AX, 0
   JGE @START
   PUSH AX
   MOV AH, 2
   MOV DL, "-"
   INT 21H
   POP AX
   NEG AX
   @START:
   XOR CX, CX
 MOV BX, 10
 @OUTPUT:
   XOR DX, DX
   DIV BX
   PUSH DX
   INC CX
   OR AX, AX
 JNE @OUTPUT
 MOV AH, 2
@DISPLAY:
POP DX
OR DL, 30H
INT 21H
LOOP @DISPLAY
 POP DX
 POP CX
 POP BX
 RET
OUTDEC ENDP
END MAIN
```

Question 2:

Write a program to find the greatest common divisor (GCD) of two integers M and N, according to the following algorithm:

*** Divide M by N, getting quotient Q and remainder R.**

*** If R = 0 then stop. N is the GCD of M and N.**

*** If R <> 0 replace M by N, N by R, and repeat step 1**

Answer:

```
MODEL SMALL
.STACK 100H
.DATA
MSG DB "ENTER THE VALUE OF M:$"
MSG1 DB 0DH,0AH,"ENTER THE VALUE OF N:$"
MSG2 DB 0DH,0AH,"THE GCD OF M AND N IS=$"
COUNT DW 0;
.CODE
MAIN PROC
   MOV AX,@DATA
   MOV DS,AX

   LEA DX,MSG
   MOV AH,9
```

```
INT 21H

CALL INDEC

PUSH AX
LEA DX,MSG1
MOV AH,9
INT 21H
CALL INDEC
MOV BX,AX
POP AX

REPEAT:
   XOR DX,DX
   DIV BX

   CMP DX,0
   JE END_LOOP

   MOV AX,BX
   MOV BX,DX
   JMP REPEAT
```

```
END_LOOP:
   LEA DX,MSG2
   MOV AH,9
   INT 21H

   MOV AX,BX
   CALL OUTDEC

   MOV AH,4CH
   INT 21H
   MAIN ENDP

INDEC PROC

   PUSH BX
   PUSH CX
   PUSH DX
   JMP READ

SKIP_BACKSPACE:
   MOV AH,2
   MOV DL,20H
   INT 21H
READ:

   XOR BX,BX
   XOR CX,CX
   XOR DX,DX

   MOV AH,1
   INT 21H

   CMP AL,"-"
   JE MINUS

   CMP AL,"+"
   JE PLUS
   JMP SKIP_INPUT

MINUS:
   MOV CH,2
   INC CL
```

```
  JMP INPUT
PLUS:

  MOV CH,2
  INC CL

INPUT:
  MOV AH,1
  INT 21H

SKIP_INPUT:


  CMP AL,0DH
  JE END_INPUT

  CMP AL,8H
  JNE NOT_BACKSPACE

  CMP CH,0
  JNE CHECK_REMOVE_MINUS
  CMP CL,0
  JE SKIP_BACKSPACE
  JMP MOVE_BACK

CHECK_REMOVE_MINUS:

  CMP CH,1
  JNE CHECK_REMOVE_PLUS

  CMP CH,1
  JE REMOVE_PLUS_MINUS

CHECK_REMOVE_PLUS:

  CMP CL,1
  JE REMOVE_PLUS_MINUS
  JMP MOVE_BACK

REMOVE_PLUS_MINUS:

  MOV AH,2
  MOV DL,20H
  INT 21H

  MOV DL,8H
  INT 21H

  JMP READ

MOVE_BACK:

  MOV AX,BX
  MOV BX,10
  DIV BX

  MOV BX,AX

  MOV AH,2
  MOV DL,20H
  INT 21H

  MOV DL,8H
  INT 21H
  XOR DX,DX
  DEC CL
  JMP INPUT

NOT_BACKSPACE:

  INC CL

  CMP AL,30H
  JL ERROR

  CMP AL,39H
  JG ERROR
  AND AX,000FH
  PUSH AX

  MOV AX,10
  MUL BX

  MOV BX,AX
  POP AX
  ADD BX,AX
  JS ERROR

  JMP INPUT

ERROR:

  MOV AH,2
  MOV DL,7H
  INT 21H

  XOR CH,CH

CLEAR:

  MOV DL,8H
  INT 21H

  MOV DL,20H
  INT 21H

  MOV DL,8H
  INT 21H


  LOOP CLEAR
  JMP READ

END_INPUT:
  CMP CH,1
  JNE EXIT
  NEG BX

EXIT:
  MOV AX,BX
  POP DX
  POP CX
  POP BX

  RET
  INDEC ENDP
  OUTDEC PROC

     PUSH BX
     PUSH CX
```

```
    PUSH DX

    CMP AX,0
    JGE START

    PUSH AX
    MOV AH,2

    MOV DL,"-"
    INT 21H
    POP AX
    NEG AX


START:
    XOR CX,CX
    MOV BX,10

OUTPUT:

    XOR DX,DX
    DIV BX
    PUSH DX
    INC CX

    OR AX,AX
    JNE OUTPUT
    MOV AH,2

DISPLAY:
    POP DX
    OR DL,30H

    INT 21H
    LOOP DISPLAY

    POP DX
    POP CX
    POP BX
    RET
    OUTDEC ENDP
  END MAIN
```