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: 01

Date of Submission: August 23,2025

Submitted by,
Name: Abrar Ahmad Chy
Student ID: 20230104063
Lab Section: B-1

Question 01: Write a program that lets the user enter time in seconds and outputs the time as hours, minutes, and seconds.

Solution:

```
.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 02: Write a program to find the greatest common divisor (GCD) of two integers.

Solution:

```
.model small
.stack 100h

.data
prompt0 db "This is a program to calculate the GCD of two inputs $"
prompt1 db 0Dh,0Ah,"Please enter integer X: $"
prompt2 db 0Dh,0Ah,"Please enter integer Y: $"
prompt3 db 0Dh,0Ah,"The GCD is: $"
intX    dw 0
intY    dw 0
gcd     dw 0

.code
main proc
    mov ax,@data
    mov ds,ax

    ; print intro
    mov ah,9
    lea dx,prompt0
    int 21h

    ; input X
    mov ah,9
    lea dx,prompt1
    int 21h
```

```
    call dec_in
    mov [intX],bx

    ; input Y
    mov ah,9
    lea dx,prompt2
    int 21h
    call dec_in
    mov [intY],bx

    ; compute gcd
    call calc_GCD
    mov bx,[gcd]

    ; show result
    mov ah,9
    lea dx,prompt3
    int 21h
    call dec_out

    mov ah,4Ch
    int 21h
main endp

; ===== INPUT DECIMAL INTO BX =====
dec_in proc
    push ax
    push dx

    xor bx,bx
    mov ah,1
    int 21h
while1:
    cmp al,0Dh
    je finis
    push ax
    mov ax,10
    mul bx
    mov bx,ax
    pop ax
    and ax,000Fh
```

```
    add bx,ax
    mov ah,1
    int 21h
    jmp while1
finis:
    pop dx
    pop ax
    ret
dec_in endp

; ===== PRINT DECIMAL IN BX =====
dec_out proc
    push ax
    push bx
    push cx
    push dx

    xor cx,cx
rept:
    mov ax,bx
    xor dx,dx
    mov bx,10
    div bx
    push dx
    inc cx
    mov bx,ax
    cmp ax,0
    jne rept

    mov ah,2
for2:
    pop dx
    or dl,30h
    int 21h
    loop for2

    pop dx
    pop cx
    pop bx
    pop ax
    ret
```

```
dec_out endp

; ===== CALCULATE GCD (Euclidean Algorithm) =====
calc_GCD proc
    mov ax,[intX]
    mov bx,[intY]

gcd_loop:
    cmp bx,0
    je done
    xor dx,dx       ; clear remainder high word
    div bx          ; ax/bx -> quotient in ax, remainder in dx
    mov ax,bx       ; new X = old Y
    mov bx,dx       ; new Y = remainder
    jmp gcd_loop

done:
    mov [gcd],ax
    ret
calc_GCD endp

end main
```

Question 03:Write a program that starts with an initially undefined byte array of maximum size 100, and lets the user insert single characters into the array in such a way that the array is always sorted in ascending order. The program should print a question mark, let the user enter a character, and display the.array With the new character Inserted. Input ends when the user hits the F.SC key. Duplicate characters should be ignored.

Solution:

```
.MODEL SMALL
.STACK 100h

.DATA
  MAX_SIZE   EQU 100
  prompt     DB '? $'
```

```
    sortedMsg   DB 0Dh,0Ah,'SORTED ARRAY: $'
    space       DB ' $'
    array       DB MAX_SIZE DUP(?)  ; array to store chars
    count       DB 0                ; number of stored elements

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

READ_LOOP:
    ; show prompt
    LEA DX, prompt
    MOV AH, 09h
    INT 21h

    ; read char
    MOV AH, 01h
    INT 21h       ; char in AL

    CMP AL, '.'   ; stop if '.'
    JE PRINT_FINAL

    ; check duplicate
    MOV CL, [count]
    XOR CH, CH
    XOR SI, SI
CHK_DUP:
    CMP CL, 0
    JE NOT_DUP
    MOV DL, [array+SI]
    CMP DL, AL
    JE SKIP_INSERT
    INC SI
    DEC CL
    JNZ CHK_DUP

NOT_DUP:
    ; find position to insert
    MOV CL, [count]
    XOR CH, CH
```

```
   XOR SI, SI
FIND_POS:
   CMP SI, CX
   JAE INSERT_HERE
   MOV DL, [array+SI]
   CMP AL, DL
   JL INSERT_HERE
   INC SI
   JMP FIND_POS

INSERT_HERE:
   ; shift right from last element to SI
   MOV CL, [count]    ; number of elements
   XOR CH, CH
   MOV DI, CX
   DEC DI          ; DI = last valid index

SHIFT_LOOP:
   CMP DI, SI
   JL PLACE_CHAR
   MOV DL, [array+DI]
   MOV [array+DI+1], DL
   DEC DI
   JMP SHIFT_LOOP

PLACE_CHAR:
   MOV [array+SI], AL
   INC BYTE PTR [count]
   JMP READ_LOOP

SKIP_INSERT:
   JMP READ_LOOP

PRINT_FINAL:
   LEA DX, sortedMsg
   MOV AH, 09h
   INT 21h

   ; print array
   MOV CL, [count]
   XOR CH, CH
```

```
   XOR SI, SI
PR_LOOP:
   CMP SI, CX
   JAE DONE
   MOV DL, [array+SI]
   MOV AH, 02h
   INT 21h

   LEA DX, space
   MOV AH, 09h
   INT 21h

   INC SI
   JMP PR_LOOP

DONE:
   MOV AH, 4Ch
   INT 21h

MAIN ENDP
END MAIN
```