**Experiment Name:**

Write a program that lets the user enter time in seconds, up to 65535 and outputs the time as hours: minutes: seconds format. Use INDEC and OUTDEC to do the I/O.

**Theory:**

The objective of this program is to show a number in hours: minutes: seconds format. INDEC and OUTDEC are needed to perform the I/O operation. For this program in assembly, While loop, CMP, Stack, INDEC, OUTDEC were used as well as the required instructions and some registers to execute the solution.

**Code:**

.MODEL SMALL

.STACK 100H

.DATA

P1 DB 'Enter the time in seconds (0 to 65535) = $'

P2 DB 0DH,0AH,'Time in hh:mm:ss format= $'

COLON DB ' : $'

.CODE

MAIN PROC

MOV AX, @DATA

MOV DS, AX

LEA DX, P1

MOV AH, 9

INT 21H

CALL INDEC

PUSH AX

LEA DX, P2

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, COLON

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, COLON

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

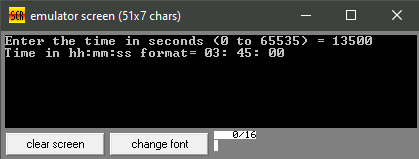
POP BX

RET

OUTDEC ENDP

END MAIN

**Output:**

****

**Discussion:**

In the above program, a number was taken as input from the user and then INDEC was called which is a procedure. The functionality of INDEC procedure is to take the input and process it like a decimal number. As in earlier assembly codes, multi-digit number was considered as an array of digit. Now using INDEC procedure, it works like a number to process with. After that, the number was pushed to the stack and then it was divided by 3600 using DIV to get the hour and if hour was greater than 9 then OUTDEC procedure was called to process that 2 digit hour and return to the output console. The above steps were repeated for both minutes and seconds also. In minutes option, it was divided by 60 using DIV. Thus in the output, using a colon(:) hours: minutes: seconds was printed on the console.