DISPLAY SYSTEM DATE AND TIME

Exp No.: 11 **Name:** S Vishakan

Date: 14-10-2020 **Reg. No:** 18 5001 196

AIM:

To write assembly language programs to perform the following system operations:

1. Display System Date
2. Display System Time

PROGRAM – 1: SYSTEM DATE:

ALGORITHM:

1. Begin.

2. Declare the data segment.
3. Initialize data segment with variables to store day, month and year.
4. Close the data segment.

5. Declare the code segment.
6. Set a preferred offset (preferably 100h)
7. Load the data segment content into AX register.
8. Transfer the contents of AX register to DS register.
9. Load 2Ah to AH register. (DOS function to obtain system date)
10. Call interrupt 21h to service the DOS function.
11. Load the offset address of variable 'day' to SI.
12. Transfer contents of DL register through SI to variable 'day'.
13. Load the offset address of variable 'month' to SI.
14. Transfer contents of DH register through SI to variable 'month'.
15. Load the offset address of variable 'year' to SI.
16. Transfer contents of CX register through SI to variable 'year'.
17. Introduce an interrupt for safe exit. (INT 21h)
18. Close the code segment.

19. End.

PROGRAM	COMMENTS
assume cs:code, ds:data	Declare code and data segment.
data segment	Initialize data segment with values.
day db 01 dup(?)	Variable to store day.
month db 01 dup(?)	Variable to store month.
year db 02 dup(?)	Variable to store year.
data ends	
code segment	Start the code segment.
org 0100h	Initialize an offset address.
start: mov ax, data	Transfer data from “data” to AX.
mov ds, ax	Transfer data from memory location AX to DS.
mov ah, 2Ah	Load 2Ah to AH (DOS code for system date function)
int 21h	Interrupt DOS with 21h to get the system date.
mov si, offset day	Load offset of variable ‘day’ to SI.
mov [si], dl	Copy to ‘day’ the value of DL through SI.
mov si, offset month	Load offset of variable ‘month’ to SI.
mov [si], dh	Copy to ‘month’ the value of DH through SI.
mov si, offset year	Load offset of variable ‘year’ to SI.
mov [si], cx	Copy to ‘year’ the value of CX through SI.
mov ah, 4ch	
int 21h	Interrupt the process with return code and exit.
code ends	
end start	

UNASSEMBLED CODE:

```
DOSBox 0.74-3, Cpu speed:    3000 cycles, Frameskip  0, Progra...

   Microsoft Object Linker V2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

Warning: No STACK segment

There was 1 error detected.

Q:\>DEBUG SYSDATE.EXE
-u
076B:0100 B86A07        MOV     AX,076A
076B:0103 8ED8          MOV     DS,AX
076B:0105 B42A          MOV     AH,2A
076B:0107 CD21          INT     21
076B:0109 BE0000        MOV     SI,0000
076B:010C 8814          MOV     [SI],DL
076B:010E BE0100        MOV     SI,0001
076B:0111 8834          MOV     [SI],DH
076B:0113 BE0200        MOV     SI,0002
076B:0116 890C          MOV     [SI],CX
076B:0118 B44C          MOV     AH,4C
076B:011A CD21          INT     21
076B:011C FF7701        PUSH    [BX+01]
076B:011F 40            INC     AX
-
```

SAMPLE I/O SNAPSHOT:

```
DOSBox 0.74-3, Cpu speed:    3000 cycles, Frameskip  0, Progra...
076B:011A CD21          INT     21
076B:011C FF7701        PUSH    [BX+01]
076B:011F 40            INC     AX
-d 076A:0000
076A:0000  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-g

Program terminated normally
-d 076A:0000
076A:0000  0E 0A E4 07 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-
```

PROGRAM – 2: SYSTEM TIME:

ALGORITHM:

1. Begin.

2. Declare the data segment.
3. Initialize data segment with variables to store hour, minute and second.
4. Close the data segment.

5. Declare the code segment.
6. Set a preferred offset (preferably 100h)
7. Load the data segment content into AX register.
8. Transfer the contents of AX register to DS register.
9. Load 2Ch to AH register. (DOS function to obtain system time)
10. Call interrupt 21h to service the DOS function.
11. Load the offset address of variable 'hour' to SI.
12. Transfer contents of CH register through SI to variable 'hour'.
13. Load the offset address of variable 'minute' to SI.
14. Transfer contents of CL register through SI to variable 'minute'.
15. Load the offset address of variable 'second' to SI.
16. Transfer contents of DH register through SI to variable 'second'.
17. Introduce an interrupt for safe exit. (INT 21h)
18. Close the code segment.

19. End.

PROGRAM	COMMENTS
assume cs:code, ds:data	Declare code and data segment.
data segment	Initialize data segment with values.
hour db 01 dup(?)	Variable to store hour.
minute db 01 dup(?)	Variable to store minute.
second db 02 dup(?)	Variable to store second.
data ends	
code segment	Start the code segment.
org 0100h	Initialize an offset address.
start: mov ax, data	Transfer data from “data” to AX.
mov ds, ax	Transfer data from memory location AX to DS.
mov ah, 2Ch	Load 2Ch to AH (DOS code for system time function)
int 21h	Interrupt DOS with 21h to get the system time.
mov si, offset hour	Load offset of variable ‘hour’ to SI.
mov [si], ch	Copy to ‘hour’ the value of CH through SI.
mov si, offset minute	Load offset of variable ‘minute’ to SI.
mov [si], cl	Copy to ‘minute’ the value of CL through SI.
mov si, offset second	Load offset of variable ‘second’ to SI.
mov [si], dh	Copy to ‘second’ the value of DH through SI.
mov ah, 4ch	
int 21h	Interrupt the process with return code and exit.
code ends	
end start	

UNASSEMBLED CODE:

```
DOSBox 0.74-3, Cpu speed:    3000 cycles, Frameskip  0, Progra...

   Microsoft Object Linker V2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

Warning: No STACK segment

There was 1 error detected.

Q:\>DEBUG SYSTIME.EXE
-u
076B:0100 B86A07        MOV     AX,076A
076B:0103 8ED8          MOV     DS,AX
076B:0105 B42C          MOV     AH,2C
076B:0107 CD21          INT     21
076B:0109 BE0000        MOV     SI,0000
076B:010C 882C          MOV     [SI],CH
076B:010E BE0100        MOV     SI,0001
076B:0111 880C          MOV     [SI],CL
076B:0113 BE0200        MOV     SI,0002
076B:0116 8834          MOV     [SI],DH
076B:0118 B44C          MOV     AH,4C
076B:011A CD21          INT     21
076B:011C FF7701        PUSH    [BX+01]
076B:011F 40            INC     AX
-
```

SAMPLE I/O SNAPSHOT:

```
DOSBox 0.74-3, Cpu speed:    3000 cycles, Frameskip  0, Progra...
076B:011A CD21          INT     21
076B:011C FF7701        PUSH    [BX+01]
076B:011F 40            INC     AX
-d 076A:0000
076A:0000  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-g

Program terminated normally
-d 076A:0000
076A:0000  12 26 07 00 00 00 00 00-00 00 00 00 00 00 00 00   .&..............
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-
```

RESULT:

The assembly level programs were written to perform the above specified system operations, namely, system date and system time and the output was verified.