Name:	Pankhania Aanandi R.
Roll No:	IT081
Batch:	I1

EXPERIMENT-10

Aim: Study of implementation of TSR:

Program 1: Active TSR using hot key combination.

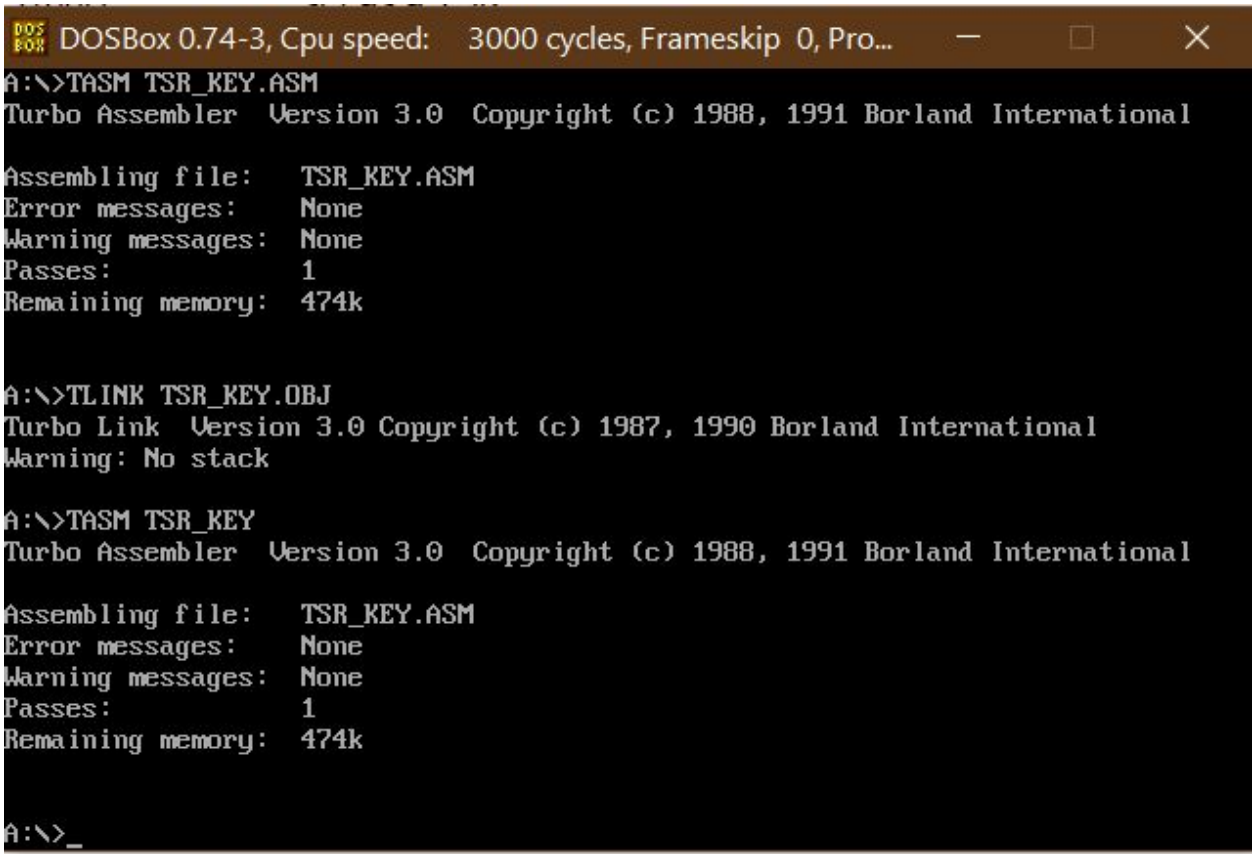
Code:

```
;a TSR program
;the ALT-K key combination is activated.
;A hot key is composed of a key scan code and a code found in memory location 0000:0417.
;The keyboard generates type9 interrupt whenever a key is typed. When intercepted with the TSR handler, it reads the keyboard code directly from I/O port 60H, which returns the keyboard scan code.


        .MODEL TINY
        .386
        .CODE
        .STARTUP
        JMP    INSTALL              ;install  VEC9

HFLAG        DB      0              ;Hot-key detected
ADD9  DD     ?                      ;old vector 9 address

KEY    DB    25H                    ;scan code for K
HMASK        DB      8              ;alternate key mask
MKEY  DB     8                      ;alternate key
SCRN  DB     300 DUP (?)            ;screen buffer
MES1  DB     'TSR IS ACTIVE'


VEC9  PROC FAR                      ;keyboard intercept
```

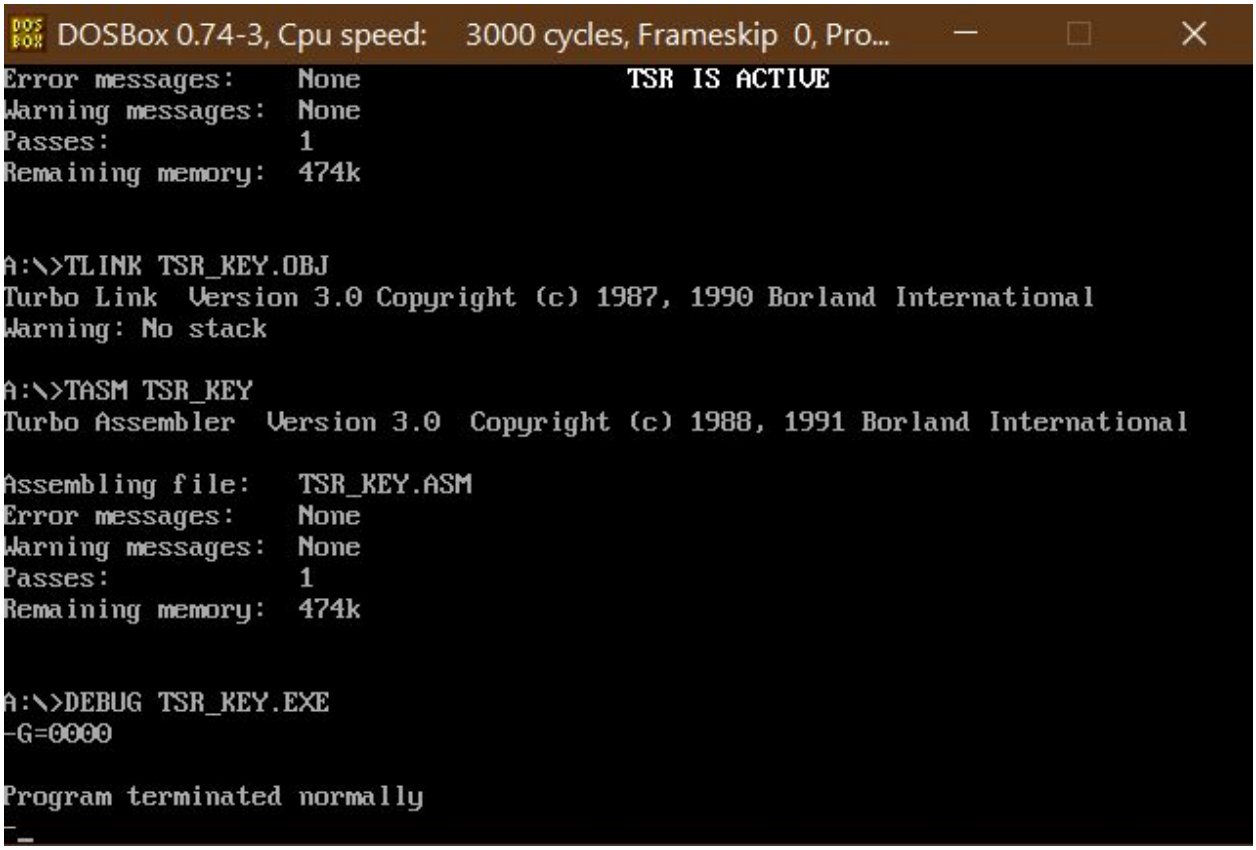
```
        STI                     ;enable interrupts
        PUSH  AX                ;save AX
        IN    AL,60H            ;get scan code
        CMP   AL,CS:KEY         ;test for K
        JNE   VEC91             ;no hot-key
        MOV   AX,0              ;address segment 0000
        PUSH  DS                ;save DS
        MOV   DS,AX
        MOV   AL,DS:[417H]      ;get shift/alternate data
        POP   DS
        AND   AL,CS:HMASK       ;isolate alternate key
        CMP   AL,CS:MKEY        ;test for alternate key
        JE    VEC93             ;if hot-key found
VEC91:

        POP   AX
        JMP   CS:ADD9           ;do normal interrupt
VEC93:                          ;if hot-key pressed
        CLI                     ;interrupts off
        IN    AL,61H            ;clear keyboard and
        OR    AL,80H            ;throw away hot key
        OUT   61H,AL
        AND   AL,7FH
        OUT   61H,AL
        MOV   AL,20H            ;reset keyboard interrupt
        OUT   20H,AL
        STI                     ;enable interrupts
        MOV   CS:HFLAG,1        ;indicate hot-key pressed
      push cx

      push di
      push si
      push ds
      push es
      cld
      mov ax,cs
      mov es,ax
      mov ax,0b800h
      mov ds,ax
      mov cx,160
      mov di,offset scrn
      mov si,0
      rep movsb
      push ds
```

```
        push es
        pop ds
        pop es
        mov di,80
        mov si,offset mes1
        mov ah,0fh
        mov cx,13
vec95: lodsb
       stosw
       loop vec95
       pop es
       pop ds
       pop si
       pop di

       pop cx
       POP     AX
          IRET

VEC9  ENDP


INSTALL:                            ;install VEC9

          MOV   AX,CS               ;load DS
          MOV   DS,AX


          MOV   AX,3509H            ;get current vector 9
          INT     21H               ;and save it
          MOV   WORD PTR ADD9,BX
          MOV   WORD PTR ADD9+2,ES

          MOV   AX,2509H
          MOV   DX,OFFSET VEC9      ;address interrupt procedure
          INT     21H               ;install vector 9

          MOV   DX,OFFSET INSTALL         ;find paragraphs
          SHR   DX,4
          INC     DX

          MOV   AX,3100H            ;set as a TSR
          INT     21H
          END
```

Compilation:



Output: //After Pressing Ctrl+Alt



Program 2: Example of Active and Passive TSR - Screensaver.

Code:

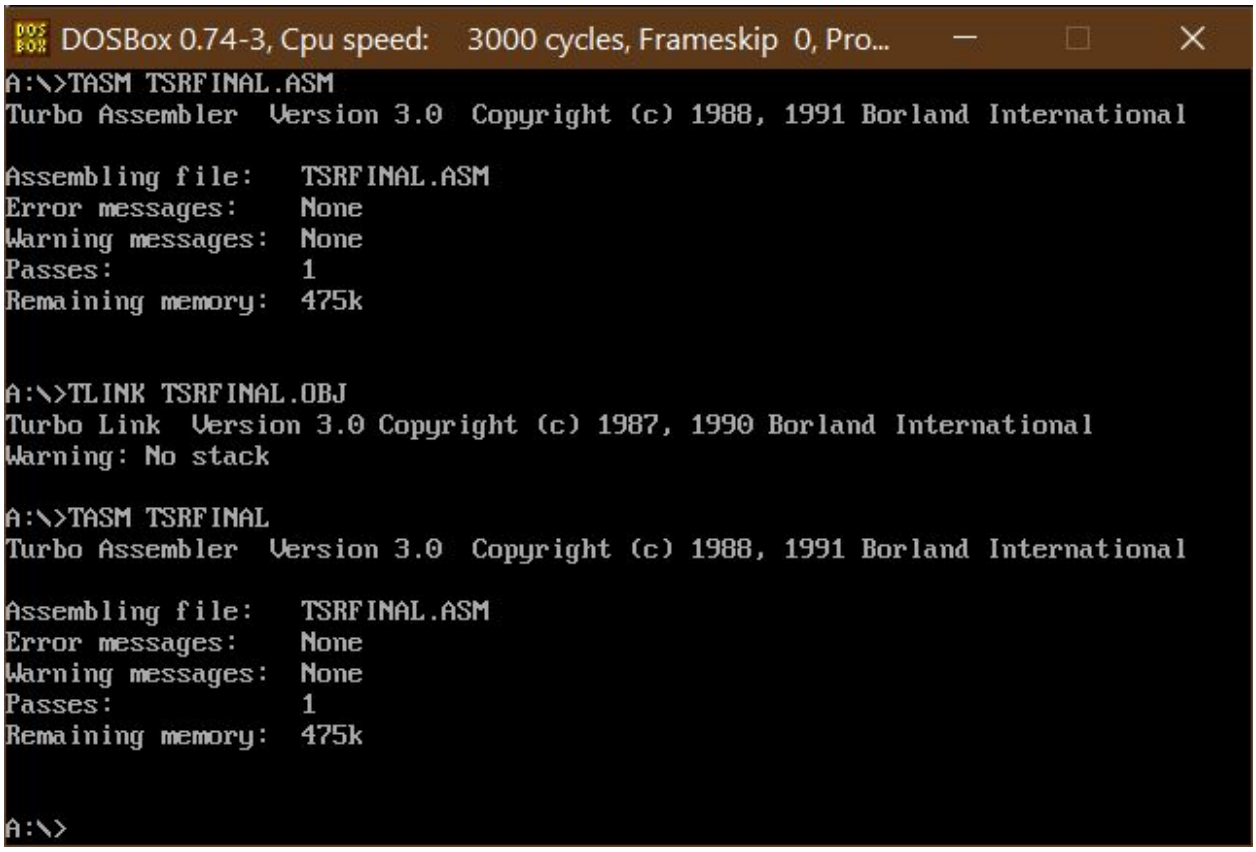
```
;Write a TSR program in 8086 ALP to implement Screen Saver. Screen Saver should get
;activated if the keyboard is idle for 7 seconds. Access the video RAM directly in your routine.
;http://books.google.co.in/books?id=zWrZY1OgTPsC&pg=PA283&lpg=PA283&dq=tsr+program;
+in+8086+with+hot+key+combination&source=bl&ots=9A_74oJXRL&sig=iqn5tQUedewU44M8
YPnDPxMP6bk&hl=en&sa=X&ei=F-b6U_zHII2jugTR84HABw&ved=0CBwQ6AEwAA#v=onepag
e&q=tsr%20program%20in%208086%20with%20hot%20key%20combination&f=false
```

```
CODE SEGMENT
        ASSUME CS:CODE,DS:CODE,ES:CODE
        ORG 100H
START : JMP BEGIN
        TIMER_IP DW ?
        TIMER_CS DW ?
        KB_IP DW ?
        KB_CS DW ?
        FLAG DB 0
        CNT DB 180
        BUFFER DW 2000 DUP(0)
TIMER:
        PUSH AX
        PUSH BX
        PUSH CX
        PUSH DX
        PUSH SI
        PUSH DI
        PUSH DS
        PUSH ES

        MOV AX,CS
        MOV DS,AX
        MOV ES,AX

        CMP FLAG,00H
        JNE TIMER_END
        DEC CNT
        JNE TIMER_END

        CLD
        MOV AX,0B800H
        MOV DS,AX
        MOV SI,0000H
        MOV DI,OFFSET BUFFER
        MOV CX,2000
        REP MOVSW

        MOV AX,0B800H
        MOV ES,AX
        MOV DI,0000H
        MOV AL,48
        MOV AH,89
```

```
        MOV CX,2000
        REP STOSW

        MOV CS:FLAG,01H
TIMER_END:
        POP ES
        POP DS
        POP DI
        POP SI
        POP DX
        POP CX
        POP BX
        POP AX
JMP DWORD PTR CS:TIMER_IP
KB:
        PUSH AX
        PUSH BX
        PUSH CX
        PUSH DX
        PUSH SI
        PUSH DI
        PUSH DS
        PUSH ES

        MOV AX,CS
        MOV DS,AX
        MOV ES,AX

        MOV CNT,180
        CMP FLAG,01
        JNE KB_END

        CLD
        MOV AX,0B800H
        MOV ES,AX
        MOV SI,OFFSET BUFFER
        MOV DI,0000H
        MOV CX,2000
        REP MOVSW

        MOV FLAG,00H
KB_END :
        POP ES
        POP DS
```

```
        POP DI
        POP SI
        POP DX
        POP CX
        POP BX
        POP AX
JMP DWORD PTR CS:KB_IP

BEGIN:
        MOV AX,CS
        MOV DS,AX
        MOV ES,AX

        MOV AH,35H
        MOV AL,08H
        INT 21H

        MOV TIMER_IP,BX
        MOV TIMER_CS,ES

        MOV AH,35H
        MOV AL,09H
        INT 21H

        MOV KB_IP,BX
        MOV KB_CS,ES

        MOV AH,25H
        MOV AL,08H
        MOV DX,OFFSET TIMER
        INT 21H
        MOV AH,25H
        MOV AL,09H
        MOV DX,OFFSET KB
        INT 21H
        MOV AH,31H
        MOV DX,OFFSET BEGIN
        MOV CL,04H
        SHR DX,CL
        INC DX
        INT 21H

CODE ENDS
END START
```

Compilation and Debugging:



Output://after 7/8 seconds: