.model large

.data

exit db 0

player\_pos dw 1760d ;position of player

arrow\_pos dw 0d ;position of arrow

arrow\_status db 0d ;0 = arrow ready to go else not

arrow\_limit dw 22d ;150d

loon\_pos dw 3860d ;3990d

loon\_status db 0d

;direction of player

;up=8, down=2

direction db 0d

state\_buf db '00:0:0:0:0:0:00:00$' ;score veriable

hit\_num db 0d

hits dw 0d

miss dw 0d

game\_over\_str dw ' ',0ah,0dh

dw ' | |',0ah,0dh

dw ' |---------------|',0ah,0dh

dw ' | ^ Score ^ |',0ah,0dh

dw ' |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|',0ah,0dh

dw ' ',0ah,0dh

dw ' ',0ah,0dh

dw ' ',0ah,0dh

dw ' ',0ah,0dh

dw ' ',0ah,0dh

dw ' ',0ah,0dh

dw ' Game Over',0ah,0dh

dw ' Press Enter to start again$',0ah,0dh

game\_start\_str dw ' ',0ah,0dh

dw ' ',0ah,0dh

dw ' ',0ah,0dh

dw ' ',0ah,0dh

dw ' ====================================================',0ah,0dh

dw ' || ||',0ah,0dh

dw ' || \* Balloon Shooting Game \* ||',0ah,0dh

dw ' || ||',0ah,0dh

dw ' ||--------------------------------------------------||',0ah,0dh

dw ' || ||',0ah,0dh

dw ' || ||',0ah,0dh

dw ' || ||',0ah,0dh

dw ' || Use up and down key to move player ||',0ah,0dh

dw ' || and space button to shoot ||',0ah,0dh

dw ' || ||',0ah,0dh

dw ' || Press Enter to start ||',0ah,0dh

dw ' || ||',0ah,0dh

dw ' || ||',0ah,0dh

dw ' ====================================================',0ah,0dh

dw '$',0ah,0dh

.code

main proc

mov ax,@data

mov ds,ax

mov ax, 0B800h

mov es,ax

jmp game\_menu ;display main menu

main\_loop: ;update logic and display everything

;check any key is pressed

mov ah,1h

int 16h ;go if pressed

jnz key\_pressed

jmp inside\_loop ;or just continue

inside\_loop: ;checking every thing

cmp miss,9 ;if baloon miss 9 times.go to game over section

jge game\_over

mov dx,arrow\_pos ;checking collitions

cmp dx, loon\_pos

je hit

cmp direction,8d ;update player position

je player\_up

cmp direction,2d ;up or down based on direction veriable

je player\_down

mov dx,arrow\_limit ;hide arrow

cmp arrow\_pos, dx

jge hide\_arrow

cmp loon\_pos, 0d ;check missed loon

jle miss\_loon

jne render\_loon

hit: ;play sound if hit

mov ah,2

mov dx, 7d

int 21h

inc hits ;update score

lea bx,state\_buf ;display score

call show\_score

lea dx,state\_buf

mov ah,09h

int 21h

mov ah,2 ;new line

mov dl, 0dh

int 21h

jmp fire\_loon ;new loon pops up

render\_loon: ;draw loon

mov cl, ' ' ;hide old loon

mov ch, 1111b

mov bx,loon\_pos

mov es:[bx], cx

sub loon\_pos,160d ;and draw new one in new position

mov cl, 15d

mov ch, 1101b

mov bx,loon\_pos

mov es:[bx], cx

cmp arrow\_status,1d ;check any arrow to rander

je render\_arrow

jne inside\_loop2

render\_arrow: ;render arrow

mov cl, ' '

mov ch, 1111b

mov bx,arrow\_pos ;hide old position

mov es:[bx], cx

add arrow\_pos,4d ;draw new position

mov cl, 26d

mov ch, 1001b

mov bx,arrow\_pos

mov es:[bx], cx

inside\_loop2:

mov cl, 125d ;draw player

mov ch, 1100b

mov bx,player\_pos

mov es:[bx], cx

cmp exit,0

je main\_loop ;end main loop

jmp exit\_game

jmp inside\_loop2

player\_up: ;hide player old position

mov cl, ' '

mov ch, 1111b

mov bx,player\_pos

mov es:[bx], cx

sub player\_pos, 160d ;set new postion of player

mov direction, 0

jmp inside\_loop2 ;it will draw in main loop

player\_down:

mov cl, ' ' ;same as player up

mov ch, 1111b ;hide old one and set new postion

mov bx,player\_pos

mov es:[bx], cx

add player\_pos,160d ;and main loop draw that

mov direction, 0

jmp inside\_loop2

key\_pressed: ;input hanaling section

mov ah,0

int 16h

cmp ah,48h ;go upKey if up button is pressed

je upKey

cmp ah, 50h

je downKey

cmp ah,39h ;go spaceKey if up button is pressed

je spaceKey

cmp ah,4Bh ;go leftKey (this is for debuging)

je leftKey

;if no key is pressed go to inside of loop

jmp inside\_loop

leftKey: ;we use it for debuging

;jmp game\_over

inc miss

lea bx,state\_buf

call show\_score

lea dx,state\_buf

mov ah,09h

int 21h

mov ah,2

mov dl, 0dh

int 21h

jmp inside\_loop

upKey: ;set player direction to up

mov direction, 8d

jmp inside\_loop

downKey:

mov direction, 2d ;set player direction to down

jmp inside\_loop

spaceKey: ;shoot a arrow

cmp arrow\_status,0

je fire\_arrow

jmp inside\_loop

fire\_arrow: ;set arrow postion in player position

mov dx, player\_pos ;so arrow fire from player postion

mov arrow\_pos, dx

mov dx,player\_pos ;when fire an arrow it also set limit

mov arrow\_limit, dx ;of arrow. where it should be hide

add arrow\_limit, 22d ;150

mov arrow\_status, 1d ;set arrow status.It prevents multiple

jmp inside\_loop ;shooting

miss\_loon:

add miss,1 ;update score

lea bx,state\_buf ;display score

call show\_score

lea dx,state\_buf

mov ah,09h

int 21h

;new line

mov ah,2

mov dl, 0dh

int 21h

jmp fire\_loon

fire\_loon: ;fire new balloon

mov loon\_status, 1d

mov loon\_pos, 3860d ;3990d

jmp render\_loon

hide\_arrow:

mov arrow\_status, 0 ;hide arrow

mov cl, ' '

mov ch, 1111b

mov bx,arrow\_pos

mov es:[bx], cx

cmp loon\_pos, 0d

jle miss\_loon

jne render\_loon

jmp inside\_loop2

;print game over screen

game\_over:

mov ah,09h

;mov dh,0

mov dx, offset game\_over\_str

int 21h

mov cl, ' ' ;hide last of screen balloon

mov ch, 1111b

mov bx,arrow\_pos

mov cl, ' ' ;hide player

mov ch, 1111b

mov bx,player\_pos

;reset value ;update veriable for start again

mov miss, 0d

mov hits,0d

mov player\_pos, 1760d

mov arrow\_pos, 0d

mov arrow\_status, 0d

mov arrow\_limit, 22d ;150d

mov loon\_pos, 3860d ;3990d

mov loon\_status, 0d

mov direction, 0d

;wait for input

input:

mov ah,1

int 21h

cmp al,13d

jne input

call clear\_screen

jmp main\_loop

game\_menu:

;game menu screen

mov ah,09h

mov dh,0

mov dx, offset game\_start\_str

int 21h

;wait for input

input2:

mov ah,1

int 21h

cmp al,13d

jne input2

call clear\_screen

lea bx,state\_buf ;display score

call show\_score

lea dx,state\_buf

mov ah,09h

int 21h

mov ah,2

mov dl, 0dh

int 21h

jmp main\_loop

exit\_game: ;end of our sweet game :)

mov exit,10d

main endp

;;--------------------------------------------------------------------;;

;; ;;

;; show score in same postion on screen ;;

;; using base pointer to get segment of veriable ;;

;; ;;

;;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;;

proc show\_score

lea bx,state\_buf

mov dx, hits

add dx,48d

mov [bx], 9d

mov [bx+1], 9d

mov [bx+2], 9d

mov [bx+3], 9d

mov [bx+4], 'H'

mov [bx+5], 'i'

mov [bx+6], 't'

mov [bx+7], 's'

mov [bx+8], ':'

mov [bx+9], dx

mov dx, miss

add dx,48d

mov [bx+10], ' '

mov [bx+11], 'M'

mov [bx+12], 'i'

mov [bx+13], 's'

mov [bx+14], 's'

mov [bx+15], ':'

mov [bx+16], dx

ret

show\_score endp

;;--------------------------------------------------------------------;;

;; ;;

;; Clear the sceen ;;

;; Just set new text mood for avoiding complexicity ;;

;; ;;

;;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;;

clear\_screen proc near

mov ah,0

mov al,3

int 10h

ret

clear\_screen endp

end main