Project code

Snake game

INCLUDE Irvine32.inc

BUFFER\_SIZE = 5000

.data

;-----------------------------------------------------------Title

whatname BYTE "Enter snake's name: ",0

namee BYTE 20 dup(?)

extending byte".io",0

;-----------------------------------------------------------wall length

xWall BYTE 105 DUP(178),0

;-----------------------------------------------------------score

strScore BYTE "Your score is: ",0

score BYTE 0

;-----------------------------------------------------------printing

strTryAgain BYTE "Try Again?  1=yes, 0=no",0

invalidInput BYTE "invalid input",0

strYouDied BYTE "you died ",0

strPoints BYTE " points scored",0

blank BYTE "                                     ",0

;-----------------------------------------------------------x-axis & y-axis

snake BYTE 4, 104 DUP(120)

xPos BYTE 45,44,43,42,41, 100 DUP(?)

yPos BYTE 15,15,15,15,15, 100 DUP(?)

xPosWall BYTE 0,0,105,105           ;position of upperLeft, lowerLeft, upperRight,lowerRignt wall

yPosWall BYTE 5,24,5,24

xCoinPos BYTE ?

yCoinPos BYTE ?

xBombPos BYTE ?

yBombPos BYTE ?

inputChar BYTE "+"

lastInputChar BYTE ?

;-----------------------------------------------------------Choose level and speed

strSpeed BYTE "Choose:    Level 1       Level 2      Level 3   ",0

stage1 BYTE "LEVEL 1",0

stage2 BYTE "LEVEL 2",0

stage3 BYTE "LEVEL 3",0

speed   DWORD 0

;-----------------------------------------------------------For displaying in the start

row BYTE ?

col BYTE ?

flag BYTE ?

.code

setit PROC

    push eax

    mov eax, red+(white\*16)

    call settextcolor

    pop eax

    ret

setit ENDP

main PROC

    call setit

    call clrscr

    call getusername

    secondmain::

    call ChooseSpeed        ;let player to choose Speed

    call clrscr

    call DrawWall ;draw walls

    ;call CreateRandomBomb

    ;call DrawBomb

    call DrawScoreboard     ;draw scoreboard

    call firstsegment

    call DisplayLevel

    mov esi,0

    mov ecx,5

    drawSnake:

        call DrawPlayer         ;draw snake(start with 5 units)

        inc esi

    loop drawSnake

    call Randomize

    call CreateRandomCoin

    call DrawCoin           ;set up finish

    call CreateRandomBomb

    call DrawBomb

    gameLoop::

        mov dl,106                      ;move cursor to coordinates

        mov dh,1

        call Gotoxy

        ; get user key input

        call ReadKey

        jz noKey                        ;jump if no key is entered

        processInput:

        mov bl, inputChar

        mov lastInputChar, bl

        mov inputChar,al                ;assign variables

        noKey:

        cmp inputChar, "p"

        je gameLoop

        cmp inputChar,"w"

        je checkTop

        cmp inputChar,"s"

        je checkBottom

        cmp inputChar,"a"

        je checkLeft

        cmp inputChar,"d"

        je checkRight

        jne gameLoop    ; reloop if no meaningful key was entered

        ; check whether can continue moving

        checkBottom:

        mov cl, yPosWall[1]

        dec cl                  ;one unit ubove the y-coordinate of the lower bound

        cmp yPos[0],cl

        jl moveDown

        je died                 ;die if crash into the wall

        checkLeft:

        cmp lastInputChar, "+"  ;check whether its the start of the game

        je dontGoLeft

        mov cl, xPosWall[0]

        inc cl

        cmp xPos[0],cl

        jg moveLeft

        je died                 ; check for left

        checkRight:

        mov cl, xPosWall[2]

        dec cl

        cmp xPos[0],cl

        jl moveRight

        je died                 ; check for right

        checkTop:

        mov cl, yPosWall[0]

        inc cl

        cmp yPos,cl

        jg moveUp

        je died             ; check for up

        moveUp:

        mov eax, speed      ;slow down the moving

        add eax, speed

        add eax,speed

        add eax, speed

        add eax,speed

        call delay

        mov esi, 0          ;index 0(snake head)

        call UpdatePlayer

        mov ah, yPos[esi]

        mov al, xPos[esi]   ;alah stores the pos of the snake's next unit

        dec yPos[esi]       ;move the head up

        call DrawPlayer

        call DrawBody

        call CheckSnake

        moveDown:           ;move down

        mov eax, speed

        add eax, speed

        add eax,speed

        add eax, speed

        add eax,speed

        call delay

        mov esi, 0

        call UpdatePlayer

        mov ah, yPos[esi]

        mov al, xPos[esi]

        inc yPos[esi]

        call DrawPlayer

        call DrawBody

        call CheckSnake

        moveLeft:           ;move left

        mov eax, speed

        call delay

        mov esi, 0

        call UpdatePlayer

        mov ah, yPos[esi]

        mov al, xPos[esi]

        dec xPos[esi]

        call DrawPlayer

        call DrawBody

        call CheckSnake

        moveRight:          ;move right

        mov eax, speed

        call delay

        mov esi, 0

        call UpdatePlayer

        mov ah, yPos[esi]

        mov al, xPos[esi]

        inc xPos[esi]

        call DrawPlayer

        call DrawBody

        call CheckSnake

    ; getting points

        checkcoin::

        mov esi,0

        mov bl,xPos[0]

        cmp bl,xCoinPos

        jne gameloop            ;reloop if snake is not intersecting with coin

        mov bl,yPos[0]

        cmp bl,yCoinPos

        jne gameloop            ;reloop if snake is not intersecting with coin

        call EatingCoin         ;call to update score, append snake and generate new coin

jmp gameLoop                    ;reiterate the gameloop

        checkBomb::

            mov esi,0

            mov bl,xPos[0]

            cmp bl,xBombPos

            jne temp ;reloop if snake is not intersecting with Bomb

            mov bl,yPos[0]

            cmp bl,yBombPos

            jne temp ;reloop if snake is not intersecting with Bomb

            call YouDied ;call to kill the snake and end the game

            ;dontChgDirection:      ;dont allow user to change direction

            ;mov inputChar, bl      ;set current inputChar as previous

            ;jmp noKey              ;jump back to continue moving the same direction

            dontGoLeft:             ;forbids the snake to go left at the begining of the game

            mov inputChar, "+"      ;set current inputChar as "+"

            jmp gameLoop            ;restart the game loop

    died::

    call YouDied

    playagn::

    call ReinitializeGame           ;reinitialise everything

    exitgame::

    exit

INVOKE ExitProcess,0

main ENDP

DisplayLevel PROC

    mov al, flag

    mov dl,99

    mov dh,1

    call gotoxy

    cmp al,1                ;input validation

    je l1

    cmp al, 2

    je l2

    cmp al,3

    je l3

    l1:

        mov edx,offset stage1

        jmp outside

    l2:

        mov edx,offset stage2

        jmp outside

    l3:

        mov edx,offset stage3

        jmp outside

    outside:

        call writestring

    ret

DisplayLevel ENDP

firstsegment PROC

call setit

mov dl,50

mov dh,1                    ;username coordinates

call gotoxy

mov edx,offset namee

call writestring

mov edx, offset extending

call writestring

mov dl,99

mov dh,1

call gotoxy

mov eax,speed

cmp eax,100

je lvl1

cmp eax,50

je lvl2

cmp eax,9

je lvl3

lvl1:

    mov edx,offset stage1

    call writestring

    jmp firstsegmentend

lvl2:

    mov edx,offset stage2

    call writestring

    jmp firstsegmentend

lvl3:

    mov edx,offset stage3

    call writestring

    jmp firstsegmentend

firstsegmentend:

mov dl,0

mov dh,4

call gotoxy

mov ecx,106

line:

    mov al,'='

    call writechar

    inc dl

    call gotoxy

loop line

ret

firstsegment ENDP

getusername PROC

call setit

mov dl,45

mov dh,16

call gotoxy

mov edx,offset whatname

call writestring

mov edx,offset namee

mov ecx,20

call readstring

ret

getusername ENDP

DrawWall PROC                   ;procedure to draw wall

    call setit

    mov dl,xPosWall[0]

    mov dh,yPosWall[0]

    call Gotoxy

    mov edx,OFFSET xWall

    call WriteString            ;draw upper wall

    mov dl,xPosWall[1]

    mov dh,yPosWall[1]

    call Gotoxy

    mov edx,OFFSET xWall

    call WriteString            ;draw lower wall

    mov dl, xPosWall[2]

    mov dh, yPosWall[2]

    mov eax,178

    inc yPosWall[3]

    L11:

    call Gotoxy

    call WriteChar

    inc dh

    cmp dh, yPosWall[3]         ;draw right wall

    jl L11

    mov dl, xPosWall[0]

    mov dh, yPosWall[0]

    mov eax,178

    L12:

    call Gotoxy

    call WriteChar

    inc dh

    cmp dh, yPosWall[3]         ;draw left wall

    jl L12

    ret

DrawWall ENDP

DrawScoreboard PROC             ;procedure to draw scoreboard

    call setit

    mov dl,2

    mov dh,1

    call Gotoxy

    mov edx,OFFSET strScore     ;print string that indicates score

    call WriteString

    mov eax,"0"

    call WriteChar              ;scoreboard starts with 0

    ret

DrawScoreboard ENDP

ChooseSpeed PROC            ;procedure for player to choose speed

    call setit

    mov edx,0

    mov dl,45

    mov dh,18

    call Gotoxy

    mov edx,OFFSET strSpeed ; prompt to enter integers (1,2,3)

    call WriteString

    mov esi, 40             ; milisecond difference per speed level

    mov eax,0

    mov dl,52

    mov dh,18

    call Gotoxy

    call readInt

    cmp ax,1                ;input validation

    jl invalidspeed

    je level1

    cmp ax, 3

    jg invalidspeed

    je level3

    cmp ax,2

    je level2

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

    level1:

        mov esi,50

        mul esi

        mov speed, eax

        mov flag, 1

        jmp exitproc

    level2:

        mov esi,13

        mul esi

        mov speed, eax

        mov flag, 2

        jmp exitproc

    level3:

        mov esi,4

        mul esi

        mov speed, eax

        mov flag, 3

        jmp exitproc

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

    exitproc:

    ret

    invalidspeed:           ;jump here if user entered an invalid number

    mov dl,45

    mov dh,19

    call Gotoxy

    mov edx, OFFSET invalidInput        ;print error message

    call WriteString

    mov ax, 1000

    call delay

    mov dl,45

    mov dh,19

    call Gotoxy

    mov edx, OFFSET blank               ;erase error message after 1 sec of delay

    call writeString

    call ChooseSpeed                    ;call procedure for user to choose again

    ret

ChooseSpeed ENDP

DrawPlayer PROC         ; draw player at (xPos,yPos)

    call setit

    mov dl,xPos[esi]

    mov dh,yPos[esi]

    call Gotoxy

    mov dl, al          ;temporarily save al in dl

    mov al, namee[esi]

    cmp al, 32  ;checking with space

    jl darkchar

    jmp afterwards

    darkchar:

        mov al, 43      ;after the snakes name finishes

    afterwards:

    call WriteChar

    mov al, dl

    ret

DrawPlayer ENDP

UpdatePlayer PROC       ; erase player at (xPos,yPos)

    call setit

    mov dl, xPos[esi]

    mov dh,yPos[esi]

    call Gotoxy

    mov dl, al          ;temporarily save al in dl

    mov al, " "

    call WriteChar

    mov al, dl

    ret

UpdatePlayer ENDP

DrawCoin PROC                       ;procedure to draw coin

    ;call setit

    mov eax,green (white \* 16)

    call SetTextColor               ;set color to yellow for coin

    mov dl,xCoinPos

    mov dh,yCoinPos

    call Gotoxy

    mov al,4

    call WriteChar

    ;mov eax,white (black \* 16)     ;reset color to black and white

    ;call SetTextColor

    ret

DrawCoin ENDP

DrawBomb PROC

    ;call setit

    mov eax,black+(white\*16)

    call SetTextColor

    mov dl,xBombPos

    mov dh,yBombPos

    call Gotoxy

    mov al,232

    call WriteChar

ret

DrawBomb ENDP

CreateRandomCoin PROC               ;procedure to create a random coin

    call setit

    mov eax,49

    call RandomRange    ;0-49

    add eax, 35         ;35-84

    mov xCoinPos,al

    mov eax,17

    call RandomRange    ;0-17

    add eax, 6          ;6-23

    mov yCoinPos,al

    mov ecx, 5

    add cl, score               ;loop number of snake unit

    mov esi, 0

checkCoinXPos:

    movzx eax,  xCoinPos

    cmp al, xPos[esi]

    je checkCoinYPos            ;jump if xPos of snake at esi = xPos of coin

    continueloop:

    inc esi

loop checkCoinXPos

    ret                         ; return when coin is not on snake

    checkCoinYPos:

    movzx eax, yCoinPos

    cmp al, yPos[esi]

    jne continueloop            ; jump back to continue loop if yPos of snake at esi != yPos of coin

    call CreateRandomCoin       ; coin generated on snake, calling function again to create another set of coordinates

CreateRandomCoin ENDP

CreateRandomBomb PROC

    call setit

    mov eax,49

    call RandomRange ;0-49

    add eax,35 ;35-84

    mov xBombPos,al

    mov eax,17

    call RandomRange ;0-17

    add eax, 6 ;6-23

    mov yBombPos,al

    mov ecx, 5

    mov esi, 0

    checkBombXPos:

    movzx eax, xBombPos

    cmp al, xPos[esi]

    je checkBombYPos ;jump if xPos of snake at esi = xPos of Bomb

    continueloop:

    inc esi

    loop checkBombXPos

    ret ; return when Bomb is not on snake

    checkBombYPos:

        movzx eax, yBombPos

        cmp al, yPos[esi]

        jne continueloop ; jump back to continue loop if yPos of snake at esi != yPos of Bomb

    call CreateRandomBomb ; Bomb generated on snake, calling function again to create another set of coordinates

CreateRandomBomb ENDP

CheckSnake PROC ;check whether the snake head collides w its body

        jmp checkBomb

        temp::

        mov al, xPos[0]

        mov ah, yPos[0]

        mov esi,4 ;start checking from index 4(5th unit)

        mov ecx,1

        add cl,score

    checkXposition:

        cmp xPos[esi], al ;check if xpos same ornot

        je XposSame

        contloop:

        inc esi

    loop checkXposition

        jmp checkcoin

        XposSame: ; if xpos same, check for ypos

        cmp yPos[esi], ah

        ;je died ;if collides, snake dies

        jmp contloop

CheckSnake ENDP

DrawBody PROC               ;procedure to print body of the snake

    call setit

        mov ecx, 4

        add cl, score       ;number of iterations to print the snake body n tail

        printbodyloop:

        inc esi             ;loop to print remaining units of snake

        call UpdatePlayer

        mov dl, xPos[esi]

        mov dh, yPos[esi]   ;dldh temporarily stores the current pos of the unit

        mov yPos[esi], ah

        mov xPos[esi], al   ;assign new position to the unit

        mov al, dl

        mov ah,dh           ;move the current position back into alah

        call DrawPlayer

        cmp esi, ecx

        jl printbodyloop

    ret

DrawBody ENDP

EatingCoin PROC

    call setit

    ; snake is eating coin

    inc score

    mov ebx,4

    add bl, score

    mov esi, ebx

    mov ah, yPos[esi-1]

    mov al, xPos[esi-1]

    mov xPos[esi], al       ;add one unit to the snake

    mov yPos[esi], ah       ;pos of new tail = pos of old tail

    cmp xPos[esi-2], al     ;check if the old tail and the unit before is on the yAxis

    jne checky              ;jump if not on the yAxis

    cmp yPos[esi-2], ah     ;check if the new tail should be above or below of the old tail

    jl incy

    jg decy

    incy:                   ;inc if below

    inc yPos[esi]

    jmp continue

    decy:                   ;dec if above

    dec yPos[esi]

    jmp continue

    checky:                 ;old tail and the unit before is on the xAxis

    cmp yPos[esi-2], ah     ;check if the new tail should be right or left of the old tail

    jl incx

    jg decx

    incx:                   ;inc if right

    inc xPos[esi]

    jmp continue

    decx:                   ;dec if left

    dec xPos[esi]

    continue:               ;add snake tail and update new coin

    call DrawPlayer

    call CreateRandomCoin

    call DrawCoin

    mov dl,17               ; write updated score

    mov dh,1

    call Gotoxy

    mov al,score

    call WriteInt

    ret

EatingCoin ENDP

YouDied PROC

    call setit

    mov eax, 1000

    call delay

    Call ClrScr

    mov dl, 57

    mov dh, 12

    call Gotoxy

    mov edx, OFFSET strYouDied  ;"you died"

    call WriteString

    mov dl, 56

    mov dh, 14

    call Gotoxy

    movzx eax, score

    call Writedec

    mov edx, OFFSET strPoints   ;display score

    call WriteString

    mov dl, 50

    mov dh, 18

    call Gotoxy

    mov edx, OFFSET strTryAgain

    call WriteString        ;"try again?"

    retry:

    mov dh, 19

    mov dl, 56

    call Gotoxy

    call ReadInt            ;get user input

    cmp al, 1

    je playagn              ;playagn

    cmp al, 0

    je exitgame             ;exitgame

    mov dh, 17

    call Gotoxy

    mov edx, OFFSET invalidInput    ;"Invalid input"

    call WriteString

    mov dl, 56

    mov dh, 19

    call Gotoxy

    mov edx, OFFSET blank           ;erase previous input

    call WriteString

    jmp retry                       ;let user input again

YouDied ENDP

ReinitializeGame PROC       ;procedure to reinitialize everything

    call setit

    mov xPos[0], 45

    mov xPos[1], 44

    mov xPos[2], 43

    mov xPos[3], 42

    mov xPos[4], 41

    mov yPos[0], 15

    mov yPos[1], 15

    mov yPos[2], 15

    mov yPos[3], 15

    mov yPos[4], 15         ;reinitialize snake position

    mov score,0             ;reinitialize score

    mov lastInputChar, 0

    mov inputChar, "+"          ;reinitialize inputChar and lastInputChar

    dec yPosWall[3]         ;reset wall position

    Call ClrScr

    jmp secondmain              ;start over the game

ReinitializeGame ENDP

END main