Coal Assignment 2

Name : Abdul Rehman , Haaris

Rollno : 21F-9060 21F-9103

Section : 3A

# Code:

[org 0x0100]

jmp start

counter : dd 200000

names : db ' Abdul Rehman Haaris', 0

rollno :db ' 21F-9103 21F-9060', 0

keys: db ' For Left Movement press : a && For Right Movement press : d', 0

gameover: db 'The Game Is Over!!', 0

lives : dw 0

score : dw 0

strscore: dw '0'

printstr:

push bp

mov bp, sp

push es

push ax

push cx

push si

push di

push ds

pop es

mov di, [bp+4]

mov cx, 0xffff

xor al, al

repne scasb

mov ax, 0xffff

sub ax, cx

dec ax

jz exitshort

mov cx, ax

mov ax, 0xb800

mov es, ax

mov al, 80

mul byte [bp+8]

add ax, [bp+10]

shl ax, 1

mov di,ax

mov si, [bp+4]

mov ah, [bp+6]

cld

nextchar:

lodsb

stosw

loop nextchar

exitshort:

pop di

pop si

pop cx

pop ax

pop es

pop bp

ret 8

clrscr:

push es

push ax

push cx

push di

mov ax, 0xb800

mov es, ax

xor di, di

mov ax, 0x0720

mov cx, 2000

cld

rep stosw

pop di

pop cx

pop ax

pop es

ret

game:

push bp

mov bp,sp

sub sp,12

push ax

push cx

push bx

push dx

push es

push di

mov ax,[bp+6]

mov bx,80

imul bx

mov bx,[bp+4]

add ax,bx

shl ax,1

mov di,ax

mov ax,2

mov [bp+6],ax

mov ax,0

mov [bp+4],ax

mov ax,0xb800

mov es,ax

mov cx,8

mov ax,0x05dc

paddle:

mov [es:di],ax

add di,2

dec cx

jnz paddle

mov [bp-10],di ; most right paddle loc ;;new index genet

mov ax,67

mov [bp-2],ax ;

mov ax,222

mov [bp-4],ax

mov ax,70

mov [bp-6],ax

mov ax,1000

mov [bp-8],ax

moving:

mov bx,100

mov ax,[bp-2]

mov cx,[bp-4]

imul cx

add ax,[bp-6]

mov cx,[bp-8]

idiv cx

shl dx,1 ; for word size and for even number.

mov [bp-2],dx ; fpr new rand num.

mov di,[bp-2]

mov ax,0x070f ; food

mov [es:di],ax

mov [bp-12],di ; food loc

droping:

mov ax,[bp-12]

mov di,ax

mov ax,0x0720 ; space

mov [es:di],ax ; for previous food (remove)

add di,160 ;For next line

mov ax,0x05dc ;foods ascii

mov [es:di],ax

mov [bp-12],di ;storing foods location

cmp di,3838 ;Second last lines most last index

ja check

xor ax,ax

mov ah,0x01

int 0x16 ;It gets a key stroke (If the line's passed and you dont press a key then ax remains zero)

jz delay

xor ah,ah ;so that buffers not zero

int 0x16

cmp al,0x64 ;for d

je right1

cmp al,0x61 ; for a

je left1

delay:

mov dword[counter],200000

time:

dec dword[counter]

jnz time

dec bx

jnz droping

right1:

mov cx,4

right:

mov ax,0x0720

mov di,3840

mov [es:di],ax

mov ax,[bp-10]

mov di,ax

mov ax,0x05dc

mov [es:di],ax

cmp di,3998

je droping

add di,2

mov [bp-10],di

sub di,18

mov ax,0x0720

mov [es:di],ax

dec cx

jnz right

jmp delay

check:

mov ax,[bp-10] ;Last index of paddle (Most right)

cmp di,ax

jg lifecheck ;If more than the most right then jump

sub ax,16 ;To get the most left index

cmp di,ax

jb lifecheck

mov dx,[bp+4]

add dx,1

mov [bp+4],dx

mov [score],dx

mov ax,[bp-12] ;Foods Location

mov di,ax

mov ax,0x0720

mov [es:di],ax

jmp moving

left1:

mov cx,4

left:

mov ax,0x0720

mov di,3998

mov [es:di],ax

mov ax,[bp-10]

mov di,ax

sub di,18

mov ax,0x05dc

mov [es:di],ax

cmp di,3840

je droping

add di,18

mov ax,0x0720

mov [es:di],ax

sub di,2

mov [bp-10],di

dec cx

jnz left

jmp delay

lifecheck:

mov dx ,[bp+6]

cmp dx,0

jz exit

dec dx

mov ax,[bp-12] ;Foods index

mov di,ax

mov ax,0x0720 ;Food gets deleted if it goes below the paddle

mov [es:di],ax

mov [bp+6],dx

mov [lives],dx

jmp moving

start:

call clrscr

mov ax, 0

push ax

mov ax, 0

push ax

mov ax, 5

push ax

mov ax,names

push ax

call printstr

mov ax, 0

push ax

mov ax, 5

push ax

mov ax, 5

push ax

mov ax,rollno

push ax

call printstr

mov ax, 0

push ax

mov ax, 10

push ax

mov ax, 5

push ax

mov ax,keys

push ax

call printstr

mov dword[counter],7000000

time4:

dec dword[counter]

jnz time4

mov ax,24

push ax

mov ax,38

push ax

call clrscr

call game

exit:

call clrscr

pop di

pop es

pop dx

pop bx

pop cx

pop ax

pop bp

mov ax, 38

push ax

mov ax, 12

push ax

mov ax, 6

push ax

mov ax,gameover

push ax

call printstr

mov ax,[score]

add ax,48

mov [strscore],ax

mov ax,0xb800

mov es,ax

mov al,[strscore]

mov ah,0x07

mov di,2164

mov [es:di],ax

mov ax,0x4c00

int 0x21

## output:

