Include Irvine32.inc

.data

gnd byte "  - - - ",0

;wordd byte "faris",0

lives byte ?

lifl byte "number of lives rem : ",0

ui byte "enter  a letter : ",0

lett byte ?

len byte ?

loss byte "YOU LOST",0

winn byte "YOU WON ",0

m2 byte "f","a","z","a","n","p","e","t","a","r","u","s","a","m","a","m","u","b","i","n",0

m3 byte 5 dup(?)

i byte 5

.code

main proc

;chosing the word

push eax

push ecx

push esi

push edi

push ebx

push edx

call randomize

mov eax,4

call randomrange

mul i

mov ecx,5

mov esi,eax

mov esi,OFFSET [m2]

add esi,eax

mov edi,OFFSET m3

rep movsb

mov edx, offset m3

pop edx

pop ebx

pop edi

pop esi

pop ecx

pop eax

; calculating length of hidden word

mov len,lengthof m3-1

mov eax,lengthof m3-2

push eax

;creating the ground

call creategn

player:

call creategn

;number of lives

mov dh,0

mov dl,0

call gotoxy

mov edx,offset lifl

call writestring

mov lives,lengthof m3-3

pop eax

call writeint

;winning condition

push eax

mov al,'-'

mov edi, offset gnd

mov ecx,lengthof gnd-1

repne scasb

jne win

pop eax

;getting from player

call crlf

mov edx,offset ui

call writestring

push eax

call readchar

  mov lett,al

call writechar

;cmparing letter and word

mov edi,offset m3

mov ecx,lengthof m3-1

repne scasb

;call dumpregs

jne decr

dec edi

call update

; fahad byte 20 dup(?)

jmp player

decr:

pop eax

dec eax

cmp eax,-1

je exitt

push eax

jmp player

win:

mov dh,10

mov dl,11

call gotoxy

MOV EDX,offset winn

call writestring

jmp gexit

exitt:

mov dh,10

mov dl,11

call gotoxy

mov edx,offset loss

call writestring

jmp gexit

gexit:

exit

main endp

creategn proc

;going to xy

mov dh,8

mov dl,10

call gotoxy

;setting ecx for blank spaces

push eax

;showing 1st char

mov al,[m3+0]

call writechar

;showing blank spaces

mov edx, offset gnd

call writestring

push ecx

mov eax,0

mov bx,lengthof m3-1

mov al,[m3+4]

call writechar

pop ecx

pop eax

ret

creategn endp

update proc

push eax

push esi

mov eax,0

mov al,len

call crlf

sub eax,ecx

sub eax,1

;call writeint

cmp eax,1

jl exitt

mov ebx,2

mul ebx

mov ebx,0

mov bl,lett

cmp eax,6

jle l1

jmp exitt

l1:

mov esi,offset gnd

mov [esi+eax],bl

exitt:

pop esi

pop eax

ret

update endp

end main