**Computer Organization And Assembly**

**Language**



* **PROJECT**

**Clutch-Game**

* **TEAM NAME**

**FOCUS**

* **Group Members**

|  |  |  |
| --- | --- | --- |
| **Ali Hasnain** | **18F-0134** | **C** |
| **Shaheer Khan** | **18F-0348** | **C** |

**CLUTCH**

Console based assembly game

****

# **Description:**

Entertainment is a great source of happiness and pleasure for a human mind. In our excessively occupied and frantic working routine, one must find some moments to moderate his/her stress level.

Clutch is a casual and relaxing game where the aim is to catch falling objects and score points. A game perfect for the time when you want to play something without thinking too much.

# **Features:**

* Catch falling objects to score points
* Maximum 5 lives
* Missing an object will cost you a life
* 3 levels game
* Speed of falling objects increases with increasing level
* Level increases after certain score limit

# **SOURCE CODE:**

;//NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

;//COAL SEMESTER PROJECT

;//DESCRIPTION: (CLUTCH) A BASIC CATCHING GAME

;//DEVELOPERS: ALI HASNAIN (18F-0134) , SHAHEER KHAN (18F-0348)

include Irvine32.inc

.data

scor byte"SCORE: ",0 ;String to display"Score"

lvl byte"LEVEL: ",0 ;String to display"LEVEL"

score byte 0 ;Variable to count Score

obx\_axis byte 18 ;Variable to store x coordinate of falling object

oby\_axis byte 6 ;Variable to store y coordinate of falling object

space2 byte" ",0 ;To print space

lives byte 5 ;Variable to control lives of user (max 5)

x\_axis byte 13 ;x coordinate of bucket and boundary

y\_axis byte 5 ;y coordinate of bucket and boundary

bucket\_axis byte 0 ;this variable always stores only x coordinate of moving bucket because y coordinate remains same

live\_coord byte 23 ;this varible stores the x axis of last heart (Used when user lose a life)

caption BYTE "GAME OVER",0 ;the caption of pop-up box will be "GAME OVER"

question BYTE "WOULD YOU LIKE TO REPLAY?",0 ;It will display on pop-up box to answer yes/no

level byte 1 ;To count the level

spead dd 80 ;To control the speed of object and bucket

.code

main proc

call menu

exit

main endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

;"------------------------------------FUNCTION TO MAKE BOUNDARY-----------------------------------------";

;1)Print Score String

;2)Print Hearts

;3)Print Boundary

;4)Print Level String

make\_boundary proc

.data

boundry1 byte" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_",0

space1 byte 78 dup(" "),0

.code

mov eax,YELLOW+(BLACK\*16); ---

call settextcolor; |

mov dh,4; |

mov dl,50; |-------TO PRINT SCORE STRING

call gotoxy; |

mov edx,offset scor; |

call writestring; ---

mov eax,YELLOW+(BLACK\*16); ---

call settextcolor; |

mov dh,40; |

mov dl,50; |-------TO PRINT LEVEL STRING

call gotoxy; |

mov edx,offset lvl; |

call writestring; |

call count\_level; ---

mov eax,RED+(BLACK\*16); ---

call settextcolor; |

mov dh,38; (38 is y\_axis of heart) |

mov dl,15; (15 is x\_axis of heart) |

mov ecx,5; (To print 5 hearts) |

print:; |-------TO PRINT HEARTS

call gotoxy; |

mov al,03; ("03" is the ASCII of heart) |

call writechar; |

add dl,2; (Add 2 in x\_axis to create space) |

loop print; ---

;(Printing uper part of boundary) ---

; |

mov eax,GREEN+(BLACK\*16); |

call settextcolor; |

mov dh,y\_axis ; |

mov dl,x\_axis ; |

call gotoxy; |

mov al,218; |

call writechar; |

call gotoxy; |

mov edx,offset boundry1; |

call writestring; |

inc y\_axis; |

mov ecx,30; |

; |

;(Printing Left and Right part of boundary) |

mov al,186; |

L1:; |

mov dh,y\_axis ; (y-axis) |

mov dl,x\_axis ; (x-axis) |------To Make boundary of Game

call gotoxy; |

call writechar; |

mov edx,offset space1; |

call writestring; |

call writechar; |

inc y\_axis; |

loop l1; |

; |

;(Printing Lower part of boundary) |

; |

mov dh,y\_axis ; y-axis; |

mov dl,x\_axis ; x-axis; |

call gotoxy; |

mov edx,offset boundry1; |

call writestring; |

; ---

;(As the bucket is moving 4 spaces at a time so to reach 82 (not 83 or 81),

;bucket is initially launched at 5 spaces right to the left boundary)

sub y\_axis,2

add x\_axis,5

ret

make\_boundary endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

;-------------------------------------------"FUNCTION TO MAKE BUCKET"---------------------------------------;

make\_bucket proc

.data

bucket byte "[\_\_\_\_\_\_\_]",0

.code

mov eax,YELLOW+(BLACK\*16)

call settextcolor

mov dh,y\_axis ; y-axis

mov dl,x\_axis ; x-axis

call gotoxy

mov edx,offset bucket

call writestring

ret

make\_bucket endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

;-------------------------------------------"FUNCTION FOR BUCKET MOVEMENT"----------------------------------;

movement proc

.data

space byte" ",0 ;(Space to remove bucket and then print to the updated position)

.code

again: ;(loop for infinite running until lives are 0)

call fall\_objects ;(For falling objects)

call make\_bucket ;(To make bucket)

call count\_score ;(To count score)

call count\_lives ;(To count lives)

cmp lives, 0 ;(To check if lives are 0 or not)

je game\_over ;(asks your if he wants to play again or not)

mov eax,20

call delay ;(This delay provides time to system to read character properly )

call readkey

cmp ax,4D00h ;(Compare with left key(4d00h) to move left)

jz move\_left

cmp ax,4B00h ;(Compare with right key(4B00h) to move left)

jz move\_right

jnz again ;(If any other key is pressed except left and right it will again ask for input key)

move\_left:

cmp x\_axis,82 ;("82"is the left end point of boundary,if bucket is at the end of boundary it will not move further left)

jz again

mov dh,y\_axis ; y-axis

mov dl,x\_axis ; x-axis

call gotoxy

mov edx,offset space

call writestring ;(Previous Bucket is removed)

add x\_axis,4 ;(bucket is moved 4 spaces to left)

call make\_bucket ;(Print bucket at new position)

jmp again

move\_right:

cmp x\_axis,14 ;("14" is the right end point of boundary,if bucket is at the end of boundary it will not move further right)

jz again

mov dh,y\_axis ; y-axis

mov dl,x\_axis ; x-axis

call gotoxy

mov edx,offset space

call writestring ;(Previous Bucket is removed)

sub x\_axis,4 ;(bucket is moved 4 spaces to left)

call make\_bucket ;(Print bucket at new position)

jmp again

game\_over:

call play\_again

ret

movement endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

;-------------------------------------------"FUNCTION FOR FALLING OBJECTS"---------------------------------;

fall\_objects proc

cmp oby\_axis,36 ;(compare coordinates of object with the lower boudary,

;if they are same,then it will redefine the coordinates for object

;to again fall from upper boundary)

jz redefine\_coordinates

jmp org\_coordinates

redefine\_coordinates:

call random

mov obx\_axis,al ;(Moves a random value in x\_axis of object)

mov oby\_axis,6

jmp org\_coordinates

org\_coordinates:

mov dh,oby\_axis ; y-axis

mov dl,obx\_axis ; x-axis

call gotoxy

mov al,02 ;(Ascii of falling character)

call writechar

dec obx\_axis

mov eax,spead

call delay

call gotoxy

mov edx,offset space2

call writestring

inc oby\_axis

inc obx\_axis

ret

fall\_objects endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

;-------------------------------------------"FUNCTION TO CALCULATE SCORE"---------------------------------;

count\_score proc

mov eax,0

mov ebx,0

mov al,y\_axis

cmp oby\_axis,al

jz X

jmp here

X:

mov al,obx\_axis

mov bl,x\_axis

mov ecx,9

l1:

cmp al,bl

jz count

jnz no

count:

inc score

call count\_level

call random ;When object touch the bucket it should not touch the boundary,it should fall again

mov obx\_axis,al

mov oby\_axis,6

no:

inc bl

loop l1

here:

mov dh,4 ;

mov dl,56 ;

call gotoxy

mov eax,0

mov eax,YELLOW+(BLACK\*16)

call settextcolor

mov al,score

call writedec

ret

count\_score endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

;-----------------------------------"FUNCTION TO PRODUCE RANDOME COORDINATES FOR OBJECT"--------------------;

;(15 is the left most boundary)

;(90 is the right most boundary)

;(We need a random number from 15 to 90,So in order to do that we generate a random number from 0-90

;and check if that number is less than 15 than we will add 15 to that number,in this way all the numbers

;produced by this function will be in between (15-90)

random proc

mov eax,20

call delay

mov eax,91

call RandomRange

call Randomize

cmp eax,15

jl redefine

jmp here

redefine:

add eax,15

here:

ret

random endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

;-------------------------------------------"FUNCTION TO CALCULATE LIVES"-----------------------------------;

count\_lives proc

cmp oby\_axis,36

jz minus

jmp here

minus:

dec lives

mov dh,38

mov dl,live\_coord

call gotoxy

mov edx,offset space2 ;(To remove lives)

call writestring

sub live\_coord,2

here:

ret

count\_lives endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

;"------------------------------------FUNCTION FOR MAKING MENU-----------------------------------------";

;1)Print menu

;2)Ask user for input

;3)Perform according to user`s input

menu proc

.data

str1 byte"MAIN MENU",0

str2 byte"1) START GAME",0

str3 byte"2) INSTRUCTIONS",0

str4 byte"3) ABOUT",0

str5 byte"CHOICE: ",0

str6 byte"INVALID CHOICE!",0

str7 byte"Clutch is a casual and relaxing game",0

str8 byte"where the aim is to catch falling objects",0

str9 byte"and score points. A game perfect for the time ",0

str10 byte"when you want to play something without thinking",0

str11 byte"too much.",0

str12 byte"DEVELOPERS",0

str13 byte"ALI HASNAIN (18F-0134)",0

str14 byte"SHAHEER KHAN (18F-0348)",0

str15 byte"ABOUT",0

str16 byte"-> Press left arrow to move bucket to left",0

str17 byte"-> Press right arrow to move bucket to right",0

str18 byte"-> Droping an object will cost a life",0

str19 byte"-> Maximum 5 lives",0

str20 byte"INSTRUCTIONS",0

str21 byte"-> Catch objects to score points",0

str22 byte"-> Level will increase after 10 points",0

.code

call m\_bound; (To make boundary of menu) ----

mov eax,RED+(BLACK\*16); |

call settextcolor; |

mov dh,15; |

mov dl,52; |

call gotoxy; |

mov edx,offset str1; |

call writestring; |

; |

mov eax,GRAY+(BLACK\*16); |

call settextcolor; |

mov dh,20; |

mov dl,45; |

call gotoxy; |

mov edx,offset str2; |

call writestring; |

; |

mov dh,23; |

mov dl,45; |--------TO DRAW MAIN MENU

call gotoxy; |

mov edx,offset str3; |

call writestring; |

; |

mov dh,26; |

mov dl,45; |

call gotoxy; |

mov edx,offset str4; |

call writestring; |

; |

mov dh,31; |

mov dl,52; |

call gotoxy; |

mov edx,offset str5; |

call writestring; |

mov eax,0; |

; ----

call readchar

call writechar

cmp al,49 ;(ASCII of "1"is 49)

je start

cmp al,50 ;(ASCII of "2"is 50)

je instruct

cmp al,51 ;(ASCII of "3"is 51)

je abt

jmp invalid ;(Any other character except 1,2 & 3 will be invalid)

start:

mov dh,40;

mov dl,50;

call gotoxy;

mov edx,offset space2

mov ecx,8

mov eax,RED+(BLACK\*16);

call settextcolor;

l7:; ---

mov al,249 ;(ASCII of "."is 249) |

call writechar; |

mov eax,300; |-----(To display the loading effect)

call delay; |

call writestring; |

loop l7; ---

call clrscr

call play

jmp end\_here

instruct:

mov dh,40;

mov dl,50;

call gotoxy;

mov edx,offset space2

mov ecx,8

mov eax,RED+(BLACK\*16);

call settextcolor;

l8:

mov al,249

call writechar

mov eax,100

call delay

call writestring

loop l8

call clrscr

call m\_bound

mov eax,RED+(BLACK\*16);

call settextcolor;

mov dh,15;

mov dl,52;

call gotoxy;

mov edx,offset str20;

call writestring;

mov eax,GRAY+(BLACK\*16);

call settextcolor;

mov dh,18;

mov dl,35;

call gotoxy;

mov edx,offset str16;

call writestring;

mov dh,20;

mov dl,35;

call gotoxy;

mov edx,offset str17;

call writestring;

mov dh,22;

mov dl,35;

call gotoxy;

mov edx,offset str18;

call writestring;

mov dh,24;

mov dl,35;

call gotoxy;

mov edx,offset str19;

call writestring;

mov dh,26;

mov dl,35;

call gotoxy;

mov edx,offset str21;

call writestring;

mov dh,28;

mov dl,35;

call gotoxy;

mov edx,offset str22;

call writestring;

mov dh,40;

mov dl,45;

call gotoxy;

call waitmsg

jmp end\_here

abt:

mov dh,40;

mov dl,50;

call gotoxy;

mov edx,offset space2

mov ecx,8

mov eax,RED+(BLACK\*16);

call settextcolor;

l9:

mov al,249

call writechar

mov eax,100

call delay

call writestring

loop l9

call clrscr

call m\_bound

mov eax,RED+(BLACK\*16);

call settextcolor;

mov dh,15

mov dl,52

call gotoxy

mov edx,offset str15

call writestring

mov eax,GRAY+(BLACK\*16);

call settextcolor;

mov dh,18

mov dl,32

call gotoxy

mov edx,offset str7

call writestring

mov dh,19

mov dl,32

call gotoxy

mov edx,offset str8

call writestring

mov dh,20

mov dl,32

call gotoxy

mov edx,offset str9

call writestring

mov dh,21

mov dl,32

call gotoxy

mov edx,offset str10

call writestring

mov dh,22

mov dl,32

call gotoxy

mov edx,offset str11

mov eax,RED+(BLACK\*16);

call settextcolor;

mov dh,25

mov dl,50

call gotoxy

mov edx,offset str12

call writestring

mov eax,GRAY+(BLACK\*16);

call settextcolor;

mov dh,28

mov dl,32

call gotoxy

mov edx,offset str13

call writestring

mov dh,30

mov dl,32

call gotoxy

mov edx,offset str14

call writestring

mov dh,40;

mov dl,45;

call gotoxy;

call waitmsg

jmp end\_here

invalid:

mov dh,31

mov dl,52

call gotoxy

mov edx,offset str6

call writestring

mov eax,800

call delay

jmp end\_here

end\_here:

call clrscr

mov my\_axis ,10

call menu

ret

menu endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

;"------------------------------------FUNCTION FOR MAKING MENU BOUDNARY------------------------------------";

m\_bound proc

.data

mx\_axis byte 30

my\_axis byte 10

left\_right dd 24

up\_down dd 50

.code

mov eax,RED+(BLACK\*16);

call settextcolor

;-----(Upper part)-----

mov dh,my\_axis

mov dl,mx\_axis

mov ecx,up\_down

call gotoxy

mov al,201

call writechar

inc dl

l1:

call gotoxy

mov al,205

call writechar

inc dl

loop l1

mov al,187

call writechar

;-----(Right part)-----

inc my\_axis

mov ecx,left\_right

mov dh,my\_axis

l5:

call gotoxy

mov al,186

call writechar

inc dh

loop l5

;-----(Left part)-----

mov ecx,left\_right

mov dh,my\_axis

mov dl,mx\_axis

l2:

call gotoxy

mov al,186

call writechar

inc dh

loop l2

call gotoxy

mov al,200

call writechar

inc dl

;-----(Lower part)-----

mov ecx,up\_down

l3:

call gotoxy

mov al,205

call writechar

inc dl

loop l3

mov al,188

call writechar

ret

m\_bound endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

;"------------------------------------FUNCTION FOR GAME OVER-------------------------------------";

play\_again proc

.code

;(Re-initializing the values)

mov score,0

mov obx\_axis,18

mov oby\_axis,6

mov lives,5

mov x\_axis,13

mov y\_axis,5

mov live\_coord,23

mov level, 1

mov spead, 100

mov ebx,OFFSET caption

mov edx,OFFSET question

call MsgBoxAsk

cmp eax,6

je yes

jmp no

yes:

call clrscr

mov eax,100

call delay

call play

no:

ret

play\_again endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

;------------------------------------"FUNCTION TO PLAY GAME"------------------------------------------------;

play proc

.code

call make\_boundary ;(Make Boundary of Game)

call make\_bucket ;(Make Bucket)

call movement ;(For bucket movement)

ret

play endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

;------------------------------------"FUNCTION TO COUNT LEVEL"------------------------------------------------;

count\_level proc

.data

sped1 byte"SPEEDING UP++",0

.code

mov al,score

cmp al,10

jz increase

cmp al,20

jz increase

jmp no\_increase

increase:

inc level

mov eax,spead

sub eax,30

mov spead,eax

mov eax,CYAN+(BLACK\*16);

call settextcolor;

mov dh,40;

mov dl,62;

call gotoxy;

mov edx,offset sped1

call writestring

no\_increase:

mov eax,YELLOW+(BLACK\*16);

call settextcolor;

mov dh,40;

mov dl,57;

call gotoxy;

mov al,level

call writedec

ret

count\_level endp

;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;

End

# **SCREEN SHOTS:**











