**FAST (National University of Computer & Emerging Sciences)**

**Karachi Campus**



**Project Report**

**Computer Organization & Assembly Language**

**Section: BCS-3C**

**Group Members:**

**Muhammad Zulqarnain Warsi [21K-4689]**

**Muhammad Kazim Khan [21K-3193]**

**PROJECT TITLE:**

**“WORD HUNT”**

* **Introduction**

In this project we have designed a game that challenges our user to find some hidden words in a table. The player is allowed to move in all direction from its current position in order to find a word correctly. The users are allowed as many attempts as they want until they give up.

* **Implementation, testing and Programming code:**

We will be implementing and testing the above described functions using the supportive environment of x86 assembly language and **Irvine** library.

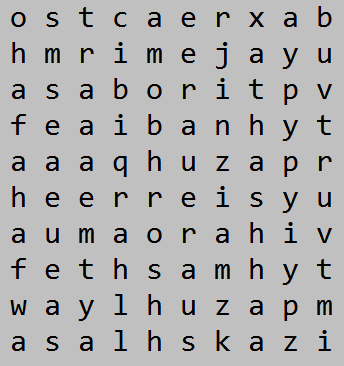
**Solution:**

When the user wants to quit the game all possible word (solution of the word puzzle) is printed with its position in table.

* **Conclusion and Future Work:**

This project can be used as a game or a part of a game and is highly user interactive as the user can easily trace words using the basic game keys that is

W A S D. For that user have to give index of first character in word and then move accordingly. The table is printed on console screen as below.



There are 5 hidden words named as KAZIM AAQIB JATIN FAHEEM and SARIM.

**Table of content**:

1. 2D arrays
2. Function of Irvine library
3. Procedures
4. Advance procedures
5. Loop
6. Conditional statements’
7. 1D array
8. Character comparator

**Project Code:**

INCLUDE irvine32.inc

.data

table byte 'o ','s ','t ','c ','a ','e ','r ','x ','a ','b '

RowSize =($-table)

byte 'h ','m ','r ','i ','m ','e ','j ','a ','y ','u '

byte 'a ','s ','a ','b ','o ','r ','i ','t ','p ','v '

byte 'f ','e ','a ','i ','b ','a ','n ','h ','y ','t '

byte 'a ','a ','a ','q ','h ','u ','z ','a ','p ','r '

byte 'h ','e ','e ','r ','r ','e ','i ','s ','y ','u '

byte 'a ','u ','m ','a ','o ','r ','a ','h ','i ','v '

byte 'f ','e ','t ','h ','s ','a ','m ','h ','y ','t '

byte 'w ','a ','y ','l ','h ','u ','z ','a ','p ','m '

byte 'a ','s ','a ','l ','h ','s ','k ','a ','z ','i '

visited byte 20 dup (0)

visrowsize =($-visited)

byte 180 dup(0)

tablekazim byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

RowSize =($-tablekazim)

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','M '

byte '\* ','\* ','\* ','\* ','\* ','\* ','K ','A ','Z ','I '

tableaaqib byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

RowSize =($-tableaaqib)

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','B ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','A ','I ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','A ','Q ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

tablesarim byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

RowSize =($-tablesarim)

byte '\* ','\* ','\* ','I ','M ','\* ','\* ','\* ','\* ','\* '

byte '\* ','S ','A ','R ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

tablefaheem byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

RowSize =($-tablefaheem)

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte 'F ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte 'A ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte 'H ','E ','E ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','M ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

tablejatin byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

RowSize =($-tablejatin)

byte '\* ','\* ','\* ','\* ','\* ','\* ','J ','A ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','I ','T ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','N ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

byte '\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* ','\* '

gap byte " ",0;

gap2 byte " ",0;

gap3 byte " ",0;

gap4 byte " ",0;

check dword ?

i dword ?

j dword ?

row dword ?

col dword ?

cnt dword ?

count dword 0

flag dword ?

ecla byte '!'

count2 dword ?

arr byte 10 dup (?)

M1 byte "Enter Row : ",0

M2 byte "Enter Column : ",0

M3 byte "Now You have to find : ",0

M4 byte "You select the Wrong alphabet try Again: ",0

M5 byte "try W,A,S,D for up,left,down and right respectively : ",0

M6 byte "Wrong choice",0

M7 byte "You Found the Word: ",0

M8 byte "Do you want to Try Another? (y/n) ",0

M9 byte "You are going Wrong... Do you want to try again? (y/n) ",0

M10 byte "Hidden words are KAZIM AAQIB SARIM JATIN FAHEEM",0

M11 byte "SOLUTIONS ARE",0

dic\_array byte 10 dup(?)

w byte "w",0

a byte "a",0

s byte "s",0

d byte "d",0

n byte "n",0

intro byte "Welcome to WORD HUNT Game!",0;

abc byte ?

count4 dword 0

holdeip dword 0

DefaultColor = black + (red \* 16)

;BlueText = red + (lightgray \* 16)

dic byte "!kazim!aaqib!sarim!jatin!faheem!hytr!",0

.code

;############################ MAIN ###############################

main PROC

mov eax,Defaultcolor

call settextcolor ;deafult have text value as black and backfround value as red

call clrscr ;to color whole screen red

call displayintro

call game

call clrscr

call displayintro

call crlf

call crlf

mov edx,offset gap4

call writestring

mov edx,offset M11

call writestring

call printkazim

call printaaqib

call printsarim

call printfaheem

call printjatin

exit

main endp

displayintro PROC

mov ecx,10

call crlf

call crlf

mov edx,offset gap

call writestring

mov edx,offset intro

call writestring

call crlf

call crlf

call crlf

mov edx,offset gap3

call writestring

mov edx,offset M10

call writestring;

call crlf

call crlf

mov edx,offset gap2

call writestring

mov i,0

L1:

mov ebx,OFFSET table

mov eax,i

mov flag,rowsize

mul flag

add ebx,eax

mov count,ecx

mov ecx,RowSize

mov esi,0

L2:

mov edx,0

mov eax,0

mov al,[ebx+esi]

call writechar

inc esi

loop L2

call crlf

mov edx,offset gap2

call writestring

inc i

mov ecx,count

loop L1

ret

displayintro endp

game PROC

above:

mov cnt,0

mov esi,offset dic\_array

mov ecx,lengthof dic\_array

clear:

mov al,[esi]

mov al,0

mov [esi],al

inc esi

loop clear

call input

mov edx,0

mov ecx,lengthof dic

mov esi,offset dic

mov dl,ecla

l3:

cmp [esi],dl

je g1

jne g2

g1:

inc count

inc esi

mov eax,0

mov eax,[esi]

mov ebx,check

cmp al,bl

jz d0

g2:

inc esi

loop l3

call crlf

call crlf

mov edx,offset M4

call writestring

jmp above

d0:

call crlf

call crlf

mov edx,offset M3

call writestring

mov edx,0

mov ecx,lengthof dic

mov dl,ecla

mov ebx,offset dic\_array

d1:

cmp [esi],dl

jne d2

je d3

d2:

mov eax,[esi]

mov [ebx],eax

inc ebx

inc esi

inc cnt

call writechar

Loop d1

d3:

call crlf

call crlf

dec cnt

mov ecx,cnt

mov ebx,1

L9:

call crlf

mov edx,offset M5

call writestring

call readchar

cmp al,d

jnz W1

mov count2,ebx

mov flag,rowsize

mov eax,row

dec eax

mul flag

mov ebx,OFFSET table

add ebx,eax

mov count,ecx

mov ecx,2

mov eax,col

mul ecx

mov esi,eax

mov al,[ebx+esi]

mov ebx, count2

cmp al,dic\_array[ebx]

jnz W2

inc ebx

call writechar

mov ecx,count

inc col

loop L9

W4:

call crlf

mov edx,offset M7

call writestring

inc cnt

mov ecx,cnt

mov ebx,0

L10:

movzx eax,dic\_array[ebx]

inc ebx

call writechar

loop L10

call crlf

call crlf

mov edx,offset M8

call writestring

call readchar

cmp al,n

jnz above

call crlf

ret

W1:

cmp al,s

jnz W5

mov count2,ebx

mov flag,rowsize

mov eax,row

mul flag

mov ebx,OFFSET table

add ebx,eax

mov count,ecx

mov ecx,2

mov eax,col

dec eax

mul ecx

mov esi,eax

mov al,[ebx+esi]

mov ebx, count2

cmp al,dic\_array[ebx]

jnz W2

inc ebx

call writechar

mov ecx,count

dec ecx

cmp ecx,0

jz W4

inc row

jmp L9

W5:

cmp al,w

jnz W6

mov count2,ebx

mov flag,rowsize

mov eax,row

dec eax

dec eax

mul flag

mov ebx,OFFSET table

add ebx,eax

mov count,ecx

mov ecx,2

mov eax,col

dec eax

mul ecx

mov esi,eax

mov al,[ebx+esi]

mov ebx, count2

cmp al,dic\_array[ebx]

jnz W2

inc ebx

call writechar

mov ecx,count

dec ecx

cmp ecx,0

jz W4

dec row

jmp L9

W6:

cmp al,a

jnz W2

mov count2,ebx

mov flag,rowsize

mov eax,row

dec eax

mul flag

mov ebx,OFFSET table

add ebx,eax

mov count,ecx

mov ecx,2

mov eax,col

dec eax

dec eax

mul ecx

mov esi,eax

mov al,[ebx+esi]

call writechar

mov ebx, count2

cmp al,dic\_array[ebx]

jnz W2

inc ebx

mov ecx,count

dec ecx

cmp ecx,0

jz W4

dec col

jmp L9

W2:

call crlf

mov edx,offset M9

call writestring

call readchar

cmp al,n

jnz above

call crlf

ret

game endp

printkazim PROC

mov ecx,10

call crlf

call crlf

call crlf

mov edx,offset gap2

call writestring

mov i,0

L1:

mov ebx,OFFSET tablekazim

mov eax,i

mov flag,rowsize

mul flag

add ebx,eax

mov count,ecx

mov ecx,RowSize

mov esi,0

L2:

mov edx,0

mov eax,0

mov al,[ebx+esi]

call writechar

inc esi

loop L2

call crlf

mov edx,offset gap2

call writestring

inc i

mov ecx,count

loop L1

ret

printkazim endp

printaaqib PROC

mov ecx,10

call crlf

call crlf

call crlf

mov edx,offset gap2

call writestring

mov i,0

L1:

mov ebx,OFFSET tableaaqib

mov eax,i

mov flag,rowsize

mul flag

add ebx,eax

mov count,ecx

mov ecx,RowSize

mov esi,0

L2:

mov edx,0

mov eax,0

mov al,[ebx+esi]

call writechar

inc esi

loop L2

call crlf

mov edx,offset gap2

call writestring

inc i

mov ecx,count

loop L1

ret

printaaqib endp

printsarim PROC

mov ecx,10

call crlf

call crlf

call crlf

mov edx,offset gap2

call writestring

mov i,0

L1:

mov ebx,OFFSET tablesarim

mov eax,i

mov flag,rowsize

mul flag

add ebx,eax

mov count,ecx

mov ecx,RowSize

mov esi,0

L2:

mov edx,0

mov eax,0

mov al,[ebx+esi]

call writechar

inc esi

loop L2

call crlf

mov edx,offset gap2

call writestring

inc i

mov ecx,count

loop L1

ret

printsarim endp

printfaheem PROC

mov ecx,10

call crlf

call crlf

call crlf

mov edx,offset gap2

call writestring

mov i,0

L1:

mov ebx,OFFSET tablefaheem

mov eax,i

mov flag,rowsize

mul flag

add ebx,eax

mov count,ecx

mov ecx,RowSize

mov esi,0

L2:

mov edx,0

mov eax,0

mov al,[ebx+esi]

call writechar

inc esi

loop L2

call crlf

mov edx,offset gap2

call writestring

inc i

mov ecx,count

loop L1

ret

printfaheem endp

printjatin PROC

mov ecx,10

call crlf

call crlf

call crlf

mov edx,offset gap2

call writestring

mov i,0

L1:

mov ebx,OFFSET tablejatin

mov eax,i

mov flag,rowsize

mul flag

add ebx,eax

mov count,ecx

mov ecx,RowSize

mov esi,0

L2:

mov edx,0

mov eax,0

mov al,[ebx+esi]

call writechar

inc esi

loop L2

call crlf

mov edx,offset gap2

call writestring

inc i

mov ecx,count

loop L1

ret

printjatin endp

input PROC

mov flag,rowsize

call crlf

call crlf

mov edx,offset gap

call writestring

mov edx,offset m1

call writestring

call readint

mov row,eax

dec eax

mul flag

mov ebx,OFFSET table

add ebx,eax

mov ecx,2

mov edx,offset gap

call writestring

mov edx,offset m2

call writestring

call readint

mov col,eax

dec eax

mul ecx

mov esi,eax

mov edx,0

mov eax,0

mov al,[ebx+esi] ;index value of table

mov check,eax

mov edx,offset check ;address of first letter

ret

input endp

;############################### PRINT JATIN END #############################

end main