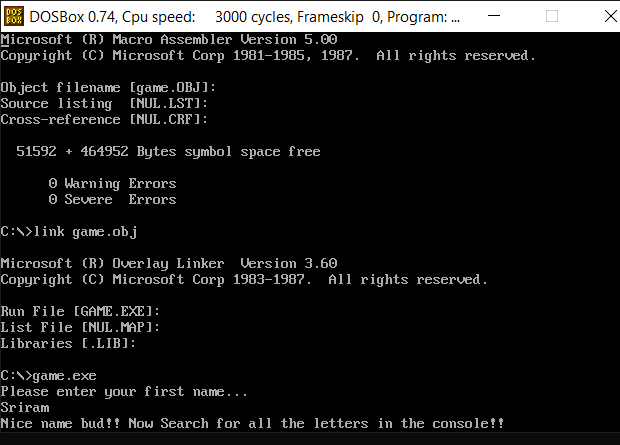
CS322 – Lab 4

Name: P. V. Sriram

Roll No.: 1801CS37

# Scrambled names!

The game I made is called Scrambled Names. The player initially has to input their first names into the console. There would be a prompt for booth asking the name and accepting one.

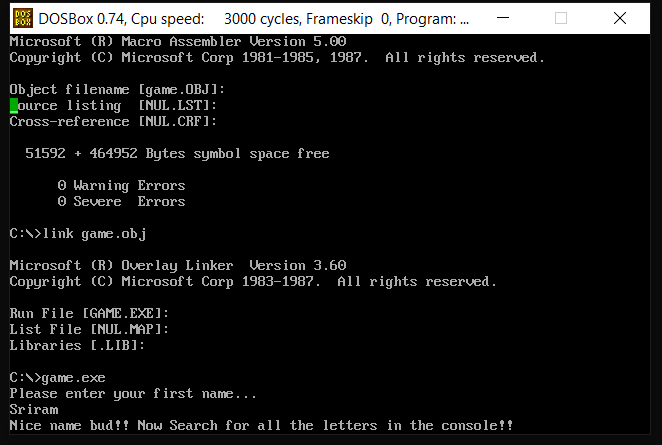


After this is done, the player has to move through the console using arrow keys and try and collect the letters in their names in the same order. When you point to cursor at the right letter, you may press enter and the console will turn green or red at that location depending on whether you picked the right letter or not. Mind that this is a case sensitive game too!

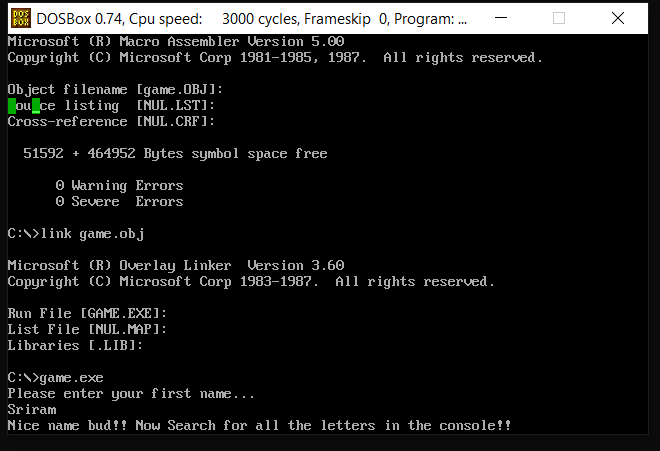
## Example

The input I gave is my own name “Sriram” and the example is based on the same.

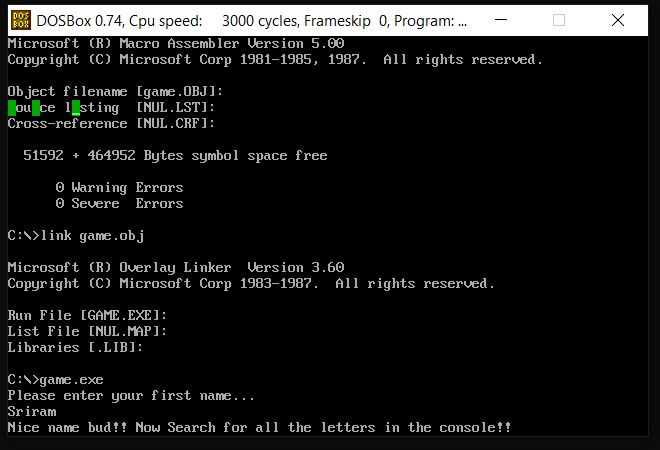
The first letter is “S” (caps), so search for one and press enter. The choice is right(green)



Next up, we have “r”, yet again the choice is right, indicated by the green color.

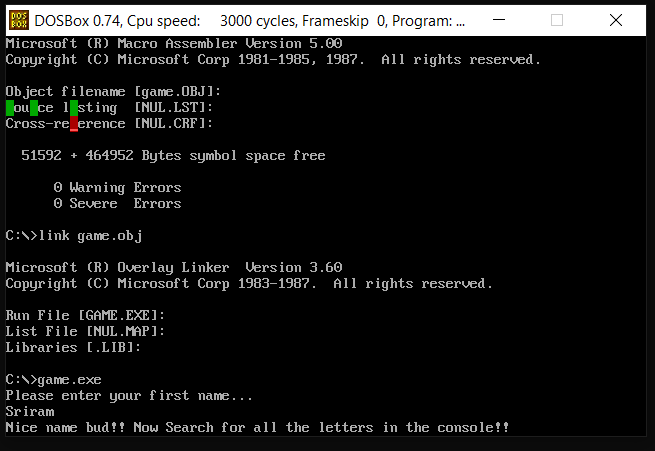


Next is “i” and the choice is right. So far so good.

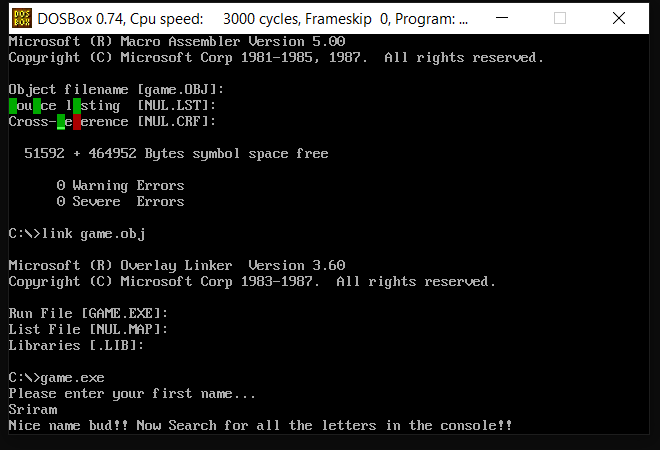


Now, if you choose a wrong letter like “f”, we can see the console indicating that

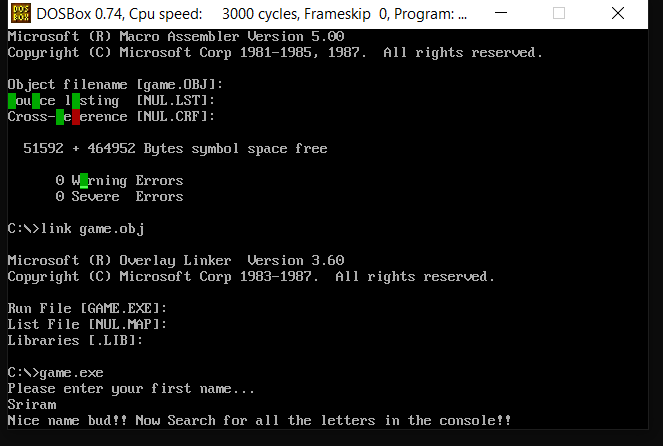
You chose the wrong letter.



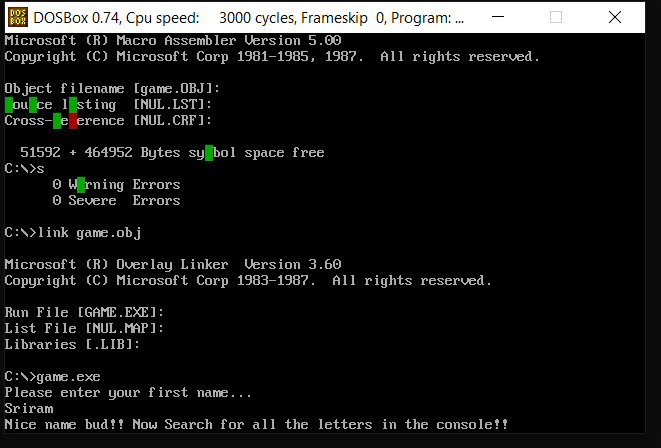
Please note that the console still expects to see an “r” and you can still continue the game.



The now runs indifferent of the wrong choice and goes as we expected.



The last letter was “m”. We chose the right letter and the execution stops and comes back to the prompt.



Note: You may quit the game anytime by pressing “esc” button.

# Code

; "Scramble names"

;AH=07h - DIRECT CHARACTER INPUT, WITHOUT ECHO

;Return: AL = character read from standard input

;Set cursor position

;10h

;AH=02h BH = Page Number, DH = Row, DL = Column

.model small

.stack 64

.data

CHARS dw 010h

ENTER\_KEY equ 0Dh

ESC\_KEY equ 1Bh ; exit

U\_KEY equ 48h

D\_KEY equ 50h

L\_KEY equ 4Bh

R\_KEY equ 4Dh

color\_red db 0CCh

color\_green db 0AAh

space db 020h

row db 0

col db 0

buffer db 16+1 dup (0) , "$" ; password buffer

buffB db 1 ; byte buffer

start db "Please enter your first name..." , 0Dh , 0Ah , "$"

PassOkay db 0Dh , 0Ah ,"Nice name bud!! Now Search for all the letters in the console!!" , "$"

.code

main: mov ax , @DATA

mov ds , ax

mov bx, ENTER\_KEY

mov cx, 0

lea dx , start

mov ah , 09h

int 21h

readUntilEnter:

mov bx , ENTER\_KEY

mov ah , 00h

int 16h

mov buffB , al

cmp bl , al ; pressed ENTER ?

je CheckLength

mov ah , 02h

mov dl , al

int 21h

mov bx , cx

mov al , buffB

mov buffer[bx] , al

inc cx

cmp cx , CHARS

jle readUntilEnter ; loop for 16 times

CheckLength:

cmp cx , CHARS

jl PassOK

PassOK:

lea si, buffer

lea dx , PassOkay

mov ah , 09h

int 21h

; cursor starting position

mov ah , 02h

mov bh , 00h

mov dx , 00h

int 10h

mov bx , ESC\_KEY

readUntilESC:

mov ah , 07h

int 21h

cmp al , U\_KEY

je UP

cmp al , D\_KEY

je DOWN

cmp al , L\_KEY

je LEFT

cmp al , R\_KEY

je RIGHT

cmp al, ENTER\_KEY

je ENTER

cmp al , ESC\_KEY

jne readUntilESC ; loop until ESC

jmp Exit

; update cursor poistion

; console resolution 80x25 (0-79 x 0-24)

UP:

cmp dh , 0

je printCursor

dec dh

jmp printCursor

DOWN:

cmp dh , 24

je printCursor

inc dh

jmp printCursor

LEFT:

cmp dl , 0

je printCursor

dec dl

jmp printCursor

RIGHT:

cmp dl , 79

je printCursor

inc dl

jmp printCursor

ENTER:

mov ah, 08h

int 10h

cmp al, [si]

je match

jne no\_match

match:

mov ah , 02h

mov bl , color\_green

mov bh , 00h

int 10h

mov ah , 09h

mov al , space ; space

push cx

mov cx , 01h

int 10h

pop cx

inc si

dec cx

jnz readUntilESC

jz exit

no\_match:

mov ah , 02h

mov bl , color\_red

mov bh , 00h

int 10h

mov ah , 09h

mov al , space ; space

push cx

mov cx , 01h

int 10h

pop cx

; print char and go back to main loop

printCursor:

mov ah , 02h

mov bh , 00h

int 10h

jmp readUntilESC

exit:

mov ah, 4ch

int 21h

end main

.end

## Functioning

* Initially, the input is read from keyboard using the readUntilEnter loop utilizing the interrupt 16. We use a max size of 16 for name in this game.
* We store this data into a buffer array of the data segment.
* After the name is entered, the movable cursor is visible on the console.
* We enter the loop readUntilEsc where the console continuously looks out for the arrow keys or enter key.
* If the input is arrow keys, we jump to the UP / DOWN / LEFT / RIGHT respectively which as their names suggest, move the cursor all over the console.
* If the input key is Enter, we move to ENTER function which maps the character on the console (using the interrupt 8) and compares it with the letter from buffer which is pointed by a pointer which increments for each correct choice.
* When the correct letter is chosen, that position on console turns green and buffer pointer increments.
* If the wrong letter is picked, the console turns red at that location and pointer stays the same.
* After all the letters are guessed, we exit from the application.