Instituto Superior de Engenharia de Coimbra
Departamento de Engenharia Informática e de Sistemas

Trabalho Prático

Licenciatura em Engenharia Informática

TAC
2016/2017

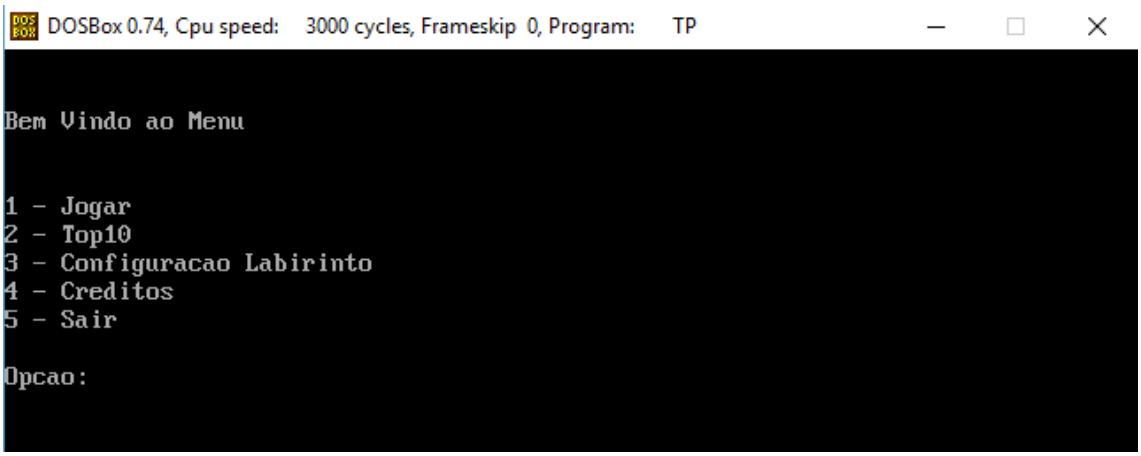
Relatório

José Hugo Sousa Silva nº21240009 – 21240009@isec.pt
José Artur Rafael Oliveira nº21240014 –21240014@isec.pt

1. Introdução

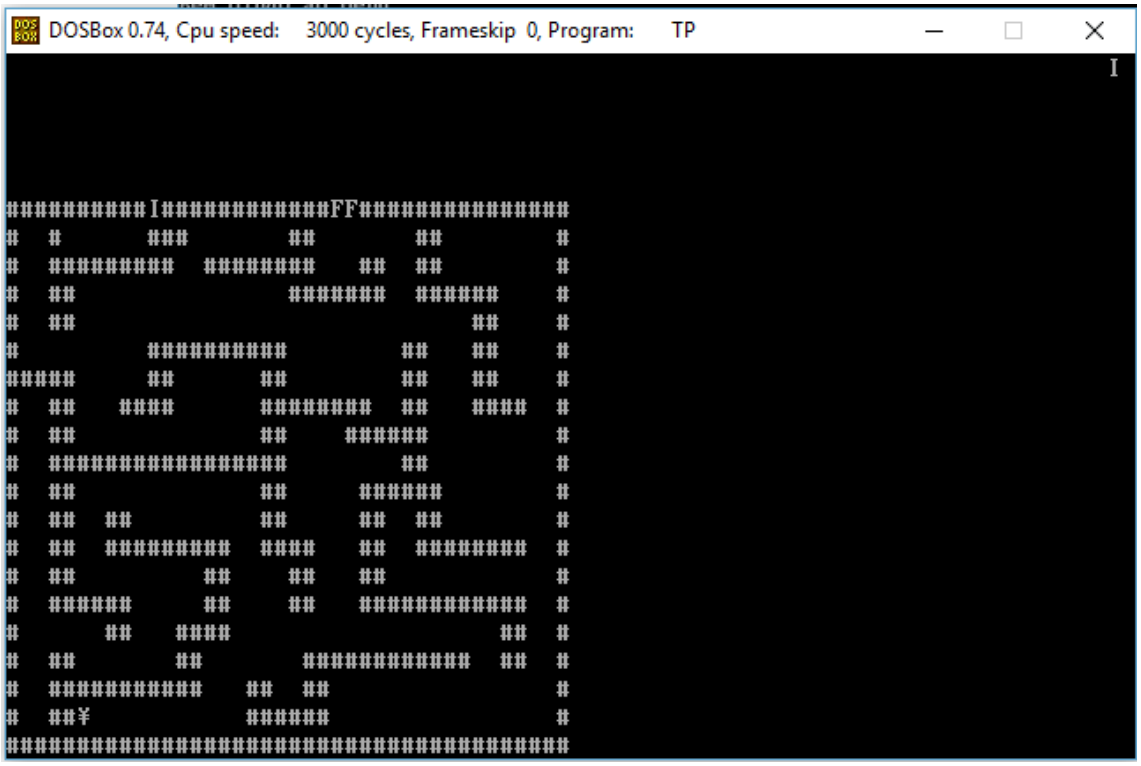
No âmbito da disciplina de Tecnologia e Arquiteturas de Computadores, foi pedido a realização de um trabalho prático, onde consiste a realização de um labirinto. O relatório consiste na explicação, e na demonstração das funções implementadas. Primeiramente irei demonstrar a interface do programa, onde possui uma interface simples e amigável com o utilizador, em seguida mostrarei o código que foi necessário para cada passo, e o porque da implementação, se possível.

2. Interface



Como é possível verificar uma interface simples, onde na primeira opção é Jogar, onde é aberto o labirinto base, e, é jogável. Na segunda opção, aparece o Top10, que não faz o que o nome indica, só guarda o valor dos segundos que o ultimo jogador fez. Na terceira opção Configuração do labirinto é possível fazer um labirinto de raiz, por paredes, fim e inicio, mas não é possível guardar, nem jogar mais tarde. Na opção 4, abre um ficheiro onde têm os créditos, como a opção indica. A opção 5 não precisa de introduções, esta opção é a opção utilizada para sair do programa/jogo.

2.1. Jogar

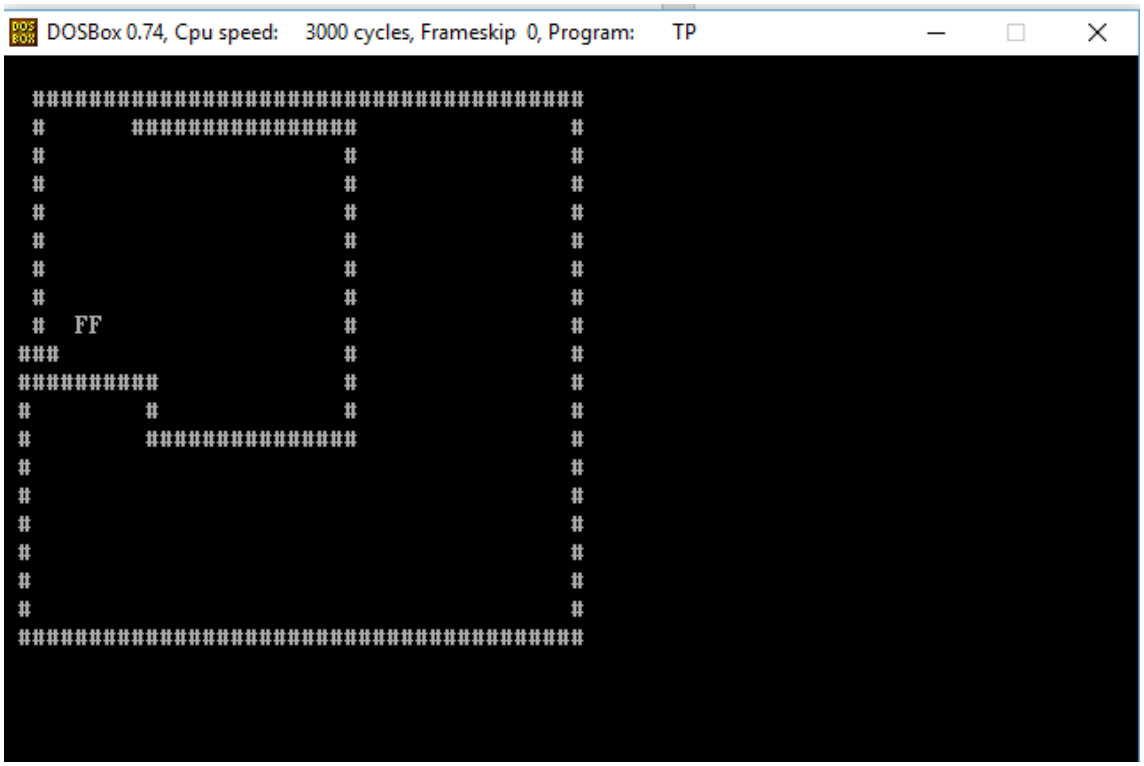


Na opção jogar como referido anteriormente, é possível, concluir o labirinto, e o avatar deteta as paredes, ao terminar eu tinha implementado aparecer o tempo no ecrã, com o sinal vitória, mas apaguei para tentar escrever em ficheiro, mas o utilizador se quiser ver o tempo que fez no menu, seleciona a opção 2 e vê o ultimo tempo que este fez. O utilizador ganha, quando chegar ao F.

2.2. Top10

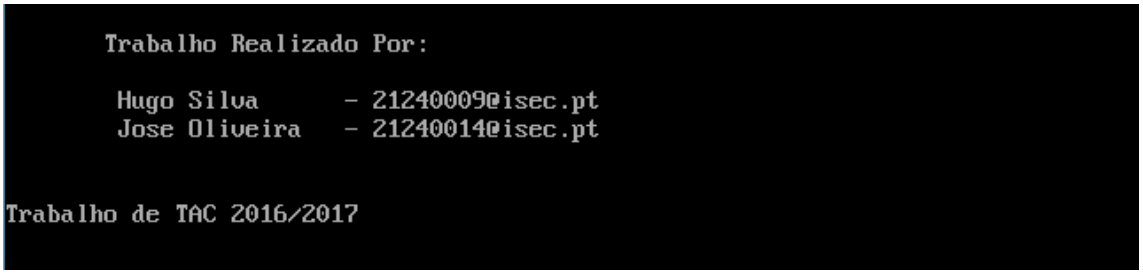
A opção Top10 correspondente há opção 2, guarda os segundos que o utilizador fez na ultima vez que jogou e ganhou.

2.3. Configuração Labirinto



Na opção de configuração de labirinto é possível fazer um labirinto, mas não é possível guardá-lo. Na configuração do labirinto, possui limites de 40x20, e é possível, acrescentar o Final(F) e o Inicio(I), e a parede. Tecla 1- Parede, Tecla 2- Fim, Tecla 3- Inicio. A Tecla 3- só pode ser pressionada uma vez, na segunda vez, passa do caracter 'I' para um espaço em branco.

2.4. Créditos



Nesta opção, simplesmente abre um ficheiro e mostra o que o ficheiro possui. Que é os créditos, onde possui os intervenientes na realização deste trabalho.

3. Código

3.1. Declaração de variáveis

```
dseg    segment para public 'data'
        ;Ler Ficheiro
        Erro_Open       db      'Erro ao tentar abrir o ficheiro$'
        Erro_Ler_Msg    db      'Erro ao tentar ler do ficheiro$'
        Erro_Close      db      'Erro ao tentar fechar o ficheiro$'
        Fich            db      'menu.txt',0
        Fichd           db      'creditos.txt',0
        Fiche           db      'labi.txt',0
        Ficho           db      'top.txt',0
        HandleFich      dw      0
        car_fich        db      ?
        ;adicionei do avatar
        string      db  "Teste prático de T.I",0
        Car         db  32  ; Guarda um caracter do Ecran
        Cor         db  7   ; Guarda os atributos de cor do caracter
        POSy        db  5   ; a linha pode ir de [1 .. 25]
        POSx        db  10  ; POSx pode ir [1..80]
        POSya       db  5   ; Posição anterior de y
        POSxa       db  10  ; Posição anterior de x
        linha       db  0   ;linha onde se encontra o ponteiro
        coluna      db  0 ;coluna onde se encontra o ponteiro
        baixos      db  0   ;seta baixo seguinte
        cimas       db  0   ;seta cima seguinte
        esquerdas   db  0   ;seta esquerda
        direitas    db  0   ;seta direita
        ;Adicionei do hms_dma
        STR12       DB      "            "  ; String para 12 digitos
        STR10       DB      "            "  ; String para 12 digitos
        NUMERO      DB      "                $",    ; String desti

        NUM_SP      db      "                $"     ; PAra apagar
        Horas       dw      0               ; Vai guardar a HORA actua
        Minutos     dw      0               ; Vai guardar os minutos a
        Segundos    dw      0               ; Vai guardar os segundos
        Old_seg     dw      0               ; Guarda os últimos segund
        Hora        dw      0               ; Vai guardar a HORA Final
        Minuto      dw      0               ; Vai guardar os minutos F
        Segundo     dw      0               ; Vai guardar os segundos
        tempo       dw      0
        ;lerlinha
        FRASE1                  db  'Acabou o jogo demorando ','$'
    buffer  db  'segundos',13,10
    contador dw 2
    msgErrorCreate  db  "Ocorreu um erro na criacao do ficheiro!$"
    msgErrorWrite   db  "Ocorreu um erro na escrita para ficheiro!$"
    msgErrorClose   db  "Ocorreu um erro no fecho do ficheiro!$"
    conf            db  0
dseg    ends
```

Declaração de Variáveis a ser utilizadas ao longo do código

3.2. Menu

```
INICIO:
    MOV AX, dseg
    MOV DS, AX
;##################
;              MAIN
;##################
jmp menu
main:
jmp sai
;##################
;MENU
MENU:
jmp abre_ficheiro
TECLAA:
call LE_TECLA
cmp al,'1'
je jogos_st
cmp al,'2'
je abre_top
cmp al,'3'
je editar_lab
cmp al,'4'
je creditos
cmp al,'5'
je sai
jmp menu
```

Código para abrir o menu, onde abre um ficheiro, com a função dada pelo professor, e, chama em seguida o Lê Tecla, também fornecido, de seguida, comparo o que está em al, com um dos caracteres (‘1’,’2’,’3’,’4’,’5’), e se for uma dessas opções vai para o indicado, se não for entre um desses números é chamado novamente o menu, e o utilizador introduz outro valor.

```
;ROTINA PARA APAGAR ECRAN
apaga_ecran proc
        mov     ax, dseg
        mov     ds,ax
        mov     ax,0B800h
        mov     es,ax
        xor     bx,bx
        mov     cx,25*80
apaga:
        mov byte ptr es:[bx],' '
        mov     byte ptr es:[bx+1],7
        inc     bx
        inc         bx
        loop        apaga
        ret
apaga_ecran endp
;########################################
;########################################
; LE UMA TECLA
LE_TECLA    PROC
        mov     ah,08h
        int     21h
        mov     ah,0
        cmp     al,0
        jne     SAI_TECLA
        mov     ah, 08h
        int     21h
        mov     ah,1
SAI_TECLA:  RET
LE_TECLA    endp
```

Função base fornecida pelo professor, onde utilizarei diversas vezes. A primeira apaga_ecran é usada para limpar o ecrã, a LE_TECLA, lê uma tecla do teclado e guarda em al.

3.3. Jogar

```
goto_xy macro       POSx,POSy
        mov     ah,02h
        mov     bh,0        ; r
        mov     dl,POSx
        mov     dh,POSy
        int     10h
endm
;################################
;funcao do jogo
jogos_st:
jmp abre_labirinto
jogos:
    call Ler_TEMPO
    mov     ax, dseg
    mov     ds,ax
    mov     ax,0B800h
    mov     es,ax
    mov POSx,5
    mov POSy,23

goto_xy POSx,POSy   ; Vai para
    mov     ah, 08h ; Guarda o
    mov     bh,0        ; numer
    int     10h
    mov     Car, al ; Guarda o
    mov     Cor, ah ; Guarda a


CICLO:  goto_xy POSxa,POSya ; \
        mov     ah, 02h
        mov     dl, Car ; Repoe
        int     21H

        goto_xy POSx,POSy   ; \
        mov     ah, 08h
        mov     bh,0        ; r
        int     10h
        mov     Car, al ; Guarc
        mov     Cor, ah ; Guarc

        goto_xy 78,0        ; M
        mov     ah, 02h ; IMPRI
        mov     dl, Car
        int     21H

        goto_xy POSx,POSy   ; \
IMPRIME:
        mov     ah, 02h
        mov     dl, 190 ; Coloc
        int     21H
        goto_xy POSx,POSy   ; \

        mov     al, POSx    ; G
        mov     POSxa, al
        mov     al, POSy    ; G
        mov     POSya, al

LER_SETA:
        call    LE_TECLA
        cmp     ah, 1
        je      ESTEND
        CMP     AL, 27  ; ESCAP
        JE      sai
        jmp     LER_SETA
```

Primeira função serve para meter o avatar na posição Posx,Posy.

Chamo o ler tempo, para calcular depois no final, o tempo que o jogador demorou a fazer o labirinto. Em seguida movo, o cursor para a posição (5,23) no ecrã. Para o inicio do labirinto base.
A seguir utilizo código base do professor, que serve para mover o avatar no ambiente gráfico. De seguida uso a função lê tecla para ler a tecla que o jogador introduz para percorrer o labirinto.

```
ESTEND:
        cmp     al,48h
        jne     BAIXO
        ;jmp    verifica_seta ; acr
        jmp     verifica_cima
        ;dec        POSy        ;ci
        ;jmp        CICLO

BAIXO:
        cmp     al,50h
        jne     ESQUERDA
        ;jmp    verifica_setab
        jmp     verifica_baixo
        ;inc    POSy        ;Baixo
        ;jmp        CICLO

ESQUERDA:
        cmp     al,4Bh
        jne     DIREITA
        jmp     verifica_esquerda
        ;dec        POSx        ;Es
        ;jmp        CICLO

DIREITA:
        cmp     al,4Dh
        jne     LER_SETA
        jmp     verifica_direita
        ;inc        POSx        ;Di
        ;jmp        CICLO


;################################
;################################

;Verifica baixo
flag:
            cmp     al, 'F'
            je      fim_jogo
            cmp     al, 20h ; C
            jne     CICLO
            inc     POSy
            jmp     CICLO


verifica_baixo:
            inc         POSy
            goto_xy     POSx,POSy
            mov         ah, 08h ; G
            mov         bh,0
            int         10h
            mov         baixos, al
            dec         POSy
            goto_xy     POSx,POSy
            jmp flag
```

```
flagc:
                cmp     al, 'F'
                je      fim_jogo
                cmp     al, 20h ; O
                jne     CICLO
                dec     POSy
                jmp     CICLO


verifica_cima:
                dec     POSy
                goto_xy POSx,POSy
                mov     ah, 08h ; Gu
                mov     bh,0
                int     10h
                mov     cimas, al
                inc     POSy
                goto_xy POSx,POSy
                jmp flagc

;verifica Esquerda
flage:
                cmp     al, 'F'
                je      fim_jogo
                cmp     al, 20h ; O
                jne     CICLO
                dec     POSx
                jmp     CICLO


verifica_esquerda:
                dec     POSx
                goto_xy POSx,POSy
                mov     ah, 08h ; Gu
                mov     bh,0
                int     10h
                mov     esquerdas,
                inc     POSx
                goto_xy POSx,POSy
                jmp flage

;verifica DIREITA
flagd:
                cmp     al, 'F'
                je      fim_jogo ; 1
                cmp     al, 20h ; O
                jne     CICLO
                inc     POSx
                jmp     CICLO


verifica_direita:
                inc     POSx
                goto_xy POSx,POSy
                mov     ah, 08h ; Gu
                mov     bh,0
                int     10h
                mov     cimas, al
                dec     POSx
                goto_xy POSx,POSy
                jmp flagd
```

Se o utilizador carregar na tecla baixo por exemplo, o que a função faz é, saltar para o verifica_baixo, e verifica a posição que está abaixo do cursor depois de verificar se é um 'F' se for termina o jogo, e salta para a função fim do jogo onde, chama outra vez a função do tempo e guarda noutras variáveis. Se não for 'F', não salta e passa para outra instrução, e verifica se é espaço se for espaço avança, e incrementamos uma posição no Posy, se não for igual a espaço, o programa salta outra vez para o ciclo. Fiz um verifica e uma flag para todos, porque cada um tem uma maneira diferente de se comportar, consoante a tecla. Mas sempre com o mesmo principio.

Na função fim de jogo, faço os cálculos do tempo ao fim do jogo, primeiramente transformo em segundos ambos os tempos (fim e inicio), de seguida guardo numa variável os valores e subtraiu o final-inicial que me dá o tempo feito pelo o jogador.

De seguida chamo a função divisão, que o que faz é transformar o numero de segundos em números separados para guardar num vetor, para depois imprimir no ficheiro, ou no ecrã. Que possuía, mas por erro meu retirei.
Exemplo: 323, fica a '3','2','3'.
Mas o numero fica ao contrário por causa do resto, neste caso não há problema, mas noutro número que não seja uma catatua, é necessário inverter a ordem, e ao passar para o vetor inverti a ordem, tendo a variável DI a decrementar, e a SI a incrementar.

```
Ler_TEMPO PROC

        PUSH AX
        PUSH BX
        PUSH CX
        PUSH DX

        PUSHF

        MOV AH, 2CH
        INT 21H

        XOR AX,AX
        MOV AL, DH
        mov Segundos, AX

        XOR AX,AX
        MOV AL, CL
        mov Minutos, AX

        XOR AX,AX
        MOV AL, CH
        mov Horas,AX

        POPF
        POP DX
        POP CX
        POP BX
        POP AX
        RET
Ler_TEMPO   ENDP
```

Função Ler_tempo utilizada em cima, para obter o tempo e guardar na variável segundos, minutos, horas. A diferença para a função ler_tempod, é que guarda em variáveis diferentes.

3.4. Top 10

Com os cálculos demonstrados em cima, passando para um buffer os números em forma de caracteres. Agora imprimir para o ficheiro, foi esta a função que o utilizei, também base do professor. Só alterando para onde vai ao fim, e, o que escreve no ficheiro.

```
Cria_ficheiro:
    mov     ah, 3ch     ; Abrir
    mov     cx, 00H     ; Defin
    lea     dx, Ficho   ; DX ap
    int     21h         ; Abre
    jnc     escreve     ; Se nã

    mov     ah, 09h
    lea     dx, msgErrorCreate
    int     21h
    jmp     sai


escreve:
    mov     bx, ax      ; Coloc
    mov     ah, 40h     ; indic
    lea     dx, STR10
    mov     cx, contador; CX fi
    int     21h
    jnc close

    ;cmp si,0
    ;jne caa
    ;lea    dx,buffer
    ;mov    cx,13
    ;int    21h
    ;jnc        close   ; Se nã



    mov     ah, 09h
    lea     dx, msgErrorWrite
    int     21h

close:
    mov     ah,3eh      ; fecha
    int     21h         ;fecha
    cmp si,0
    jnc     menu

    mov     ah, 09h
    lea     dx, msgErrorClose
    int     21h
```

3.5. Configuração do Labirinto

```
editar_lab:
        mov     ax, dseg
        mov     ds,ax
        mov     ax,0B800h
        mov     es,ax
        call        apaga_ecran
    dec     POSy        ; linha =
    dec     POSx        ; POSx = 1
CICL:   goto_xy POSx,POSy
IMPRIM: mov     ah, 02h
        mov     dl, Car
        int     21H
    goto_xy POSx,POSy
        call    LE_TECLA
        cmp     ah, 1
        je      ESTEN
        CMP     AL, 27      ; ESC
        JE      menu
ZERO:   CMP     AL, 48      ; Tec
        JNE     UM
    lala:mov    Car, 32     ;ESPA
        jmp     CICL
UM:     CMP     AL, 49      ; Tec
        JNE     DOIS
        mov     Car, '#'        ;
        jmp     CICL
DOIS:   CMP         AL, 50      ;
        JNE     TRES
        mov     Car, 'F' ;CINZA 1'
        jmp     CICL
TRES:   CMP         AL, 51      ;
        JNE     QUATRO
        cmp     conf,1
        je      lala
        mov     Car, 'I'
        inc     conf
        jmp     CICL
QUATRO: CMP     AL, 52      ; Tec
        JNE     NOVE
        mov     Car, 176        ;
        jmp     CICL
NOVE:       jmp     CICL
ESTEN:
        cmp     al,48h
        jne     BAIX
        cmp     POSy,1
        je      cicl
        dec     POSy ;cima
        jmp     CICL
BAIX:   cmp     al,50h
        jne     ESQUERD
        cmp     POSy,20
        je      cicl
        inc     POSy        ;Baix
        jmp     CICL
ESQUERD:
        cmp     al,4Bh
        jne     DIREIT
        cmp     POSx,1
        je      cicl
        dec     POSx        ;Esqu
        jmp     CICL
```

Código base do professor, com pequenas alterações, como delimitar 40x20, fazendo um cmp posx com 1 ou 40, quando pede para sair fora destes limites salta para o ciclo novamente. Para o posy acontece o mesmo, mas com limites diferentes de 1 a 20. Depois também tento delimitar as vezes que o Inicio é posto, mas sem sucesso.

```
DIREIT:
        cmp     al,4Dh
        jne     CICL
        cmp     POSx,40
        je      cicl
        inc     POSx
        jmp     CICL
```

4. Conclusão

Com a realização deste trabalho compreendi, o que é trabalhar em linguagens de baixo nível, e ver como funciona programas a baixo nível, ajudou-me bastante esta compreensão para a cadeira de P, por exemplo. Ajuda a entender ponteiros e como as linguagens e programas devem funcionar. Também aprendesse a ter alguma consistência no código, e, ver que com comandos simples é possível fazer coisas complexas e engraçadas. Este trabalho, ajudou a perceber o conceito de TAC e o que é a sua programação.