FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA
Departamento de Engenharia Electrotécnica

Trabalho Final

Microprocessadores

Turno P3

10/01/2021

Alunos:

Filipe Morais Nº 57867

João Nogueira Nº 57574

Matilde Sacavém Nº 58608

Índice

1. INTRODUÇÃO

O trabalho tem como objetivo utilizar os conhecimentos adquiridos na linguagem Assembly na unidade curricular de Microprocessadores para, com base no processador Intel 8086, desenvolvido pelos alunos na unidade curricular de Sistemas Lógicos II, criar um programa que permita distribuir os alunos por salas mantendo o seu registo, para mais tarde permitir identificar potenciais infrações (cópias de exames).

O programa deve permitir:

- Receber a lista de alunos através de um ficheiro;
- Gerar a distribuição pseudoaleatória dos alunos por sala;
- Guardar a distribuição dos alunos na sala num outro ficheiro;
- Apresentar a distribuição dos alunos na sala usando os números de aluno;
- Pesquisa os alunos com infrações numa sala e apresentar os seus números e nomes no ecrã.

Este relatório é constituído por 7 capítulos, incluindo como primeiro capítulo, esta mesma introdução, que constitui um breve resumo:

- Capítulo 2: Descrição dos requisitos específicos do trabalho, apresentando em maior detalhe o que é pretendido do mesmo, bem como as suas funcionalidades;
- Capítulo 3: Descrição do programa criado, bem como das principais funções que o constituem;
- Capítulo 4: Mostramos o programa desenvolvido e fazemos um *walkthrough* pelas variadas funcionalidades do programa;
- Capítulo 5: Apresentação as nossas reflexões científicas face ao trabalho realizado, e eventuais notas/comentários relevantes.
- Capítulo 6: Lista de referências bibliográficas.

2. DESCRIÇÃO DOS REQUISITOS

Neste capítulo apresenta-se a descrição dos requisitos específicos do trabalho, apresentando em maior detalhe o que é pretendido do mesmo, bem como as suas funcionalidades.

Ao iniciar o programa, é apresentado um menu com algumas opções as quais são possíveis selecionar utilizando o rato. Estas opções são:

- Ler Alunos - Ler de ficheiro a lista de alunos a distribuir pela sala e efetuar a distribuição;
- Consultar Localização - mostrar a distribuição dos alunos na sala;
- Guardar Localização – Guardar a localização dos alunos num ficheiro binário;
- Consultar potencial infratores- de alunos em sala para determinar se num determinado dia estiveram sentados perto de um determinado aluno e mostrar o resultado no monitor;
- Sair.

Segue-se uma descrição mais detalhada das funcionalidades do programa.

LER ALUNOS

Esta opção utiliza como base um ficheiro de tipo *.txt*, que contem uma lista de alunos, onde se encontram os seus nomes e correspondentes números (os números de aluno vão de 00 a 99, tendo o ficheiro no máximo 100 alunos), e procede à distribuição pseudoaleatória dos primeiros 60 alunos da lista, colocando os seus números num vetor de 60 posições.

CONSULTAR LOCALIZAÇÃO

Esta opção utiliza o vetor de 60 posições previamente preenchido para mostrar no ecrã a distribuição dos alunos numa sala com 12 filas e 5 colunas, perfazendo um total máximo de 60 alunos por sala.

GUARDAR LOCALIZAÇÃO

Esta opção utiliza o vetor de 60 posições previamente preenchido e copia esse vetor para um ficheiro de tipo *.cop*, tal como o nome do ficheiro *.txt* que originou a distribuição de alunos. O nome deste ficheiro é gerado com base na data do sistema e no número da sala atribuído pelo utilizador.

GUARDAR LOCALIZAÇÃO

Esta opção utiliza o ficheiro .cop previamente criado para preencher um vetor de 60 posições com os números dos alunos que estavam naquela sala. Depois o programa pede o numero do aluno infrator ao utilizador e, com essa informação, o programa acede ao ficheiro *.txt* referenciado no ficheiro *.cop* e identifica outros potenciais infratores, ou seja, os alunos que estavam em redor do aluno identificando como potenciais copiadores, devolvendo ao utilizador o nome e o número desses mesmos alunos.

SAIR

Esta opção serve para terminar o programa.

3. APRESENTAÇÃO DETALHADA DO PROGRAMA

Neste capítulo apresenta-se uma descrição do programa criado, bem como das principais funções que o constituem.

FUNÇÃO "LERALUNOS"

```
;*****************************************************************
; lerAlunos - reads each line from the file, and randomly places it in a vector
; input -
; output -
; destroys - ax, bx, cx, dx
;*****************************************************************

    proc lerAlunos

repeat:
        call setVideoMode
        call limpaArrayNums

        lea dx, perguntanomefile
        call printstring

        call showBlinkingCursor

        mov di, offset nomefilerecebido+3
        call scanf

        call setVideoMode

        lea dx, strlivro
        call printstring

        lea dx, nomefilerecebido
        call fOpen

        jc verifMouse

        mov handler0, ax
        mov numalsala, 0

numManagement:

        call tornaNum

         call randomPosition

         add numAlSala, 1

         cmp numAlSala, 60
         je  endOfLerAlunos

charCycler:
         mov bx, handler0
         lea dx, charlido
         mov cx, 1
         call fRead

         cmp ax, 0
         je endOfLerAlunos

         cmp charlido, 10
         jne charCycler

         jmp numManagement

endOfLerAlunos:
         mov bx, handler0
         call fclose

         ret

verifMouse:

         lea dx, strAgain
         call printString

         call getMousePos

          cmp cx, 210
          jb verifMouse

          cmp cx, 415
          ja verifMouse

          cmp dx, 150
          jb verifMouse

          cmp dx, 175
          ja verifMouse

          cmp bx, 1
          jne verifMouse

          jmp repeat

    endp
```

←Limpa o ecrã.
←Limpa o vetor de posições.

←Leitura do nome do ficheiro de texto escolhido pelo utilizador.

←Abertura do ficheiro escolhido pelo utilizador.

←Vai buscar um número de aluno ao ficheiro texto e coloca numa posição aleatória do vetor de posições. Quando o vetor de posições estiver cheio (tiver as 60 posições preenchidas), o ciclo acaba.

←Passa à frente todos os caracteres do ficheiro de texto que são números.

←Fecha o ficheiro de texto.

←Permite o funcionamento do botão "Repetir" que permite que o utilizador volte a colocar o nome de um ficheiro de texto para ser lido.

FUNÇÃO "CONSULOCAL"

```
;*************************************************************
; consuLocal - displays the students of one classroom in a matrix
; input -
; output -
; destroys -
;*************************************************************

    proc consuLocal

        call setvideomode
        mov charlido, 0
        mov colunas, 0
        mov linhas, 0

inicio:
        mov dl, 179
        call co

        mov al, charlido
        xor ah,ah

        lea si, arraynums
        add si, ax
        mov dl, [si]
        call TornaChar

        cmp al, 58
        jne cadeiracheia
        cmp ah, 49
        jne cadeiracheia
        mov al, 31
        mov ah, 31

cadeiracheia:

        mov dl, al
        call co
        mov dl, ah
        call co

        mov bl, charlido
        inc bl
        mov charlido, bl

        cmp dl, 0
        je returns

        mov dl, 179
        call co
        mov dl, 32
        call co

        mov bl, colunas
        inc bl
        mov colunas, bl

        cmp colunas, 5
        je backspace
        jmp inicio


returns:
        mov dh, 10
        mov dl, 25
        call setCursor
        lea dx, strRegressarPT1
        call printString


        mov dh, 11
        mov dl, 25
        call setCursor
        lea dx, strRegressarPT2
        call printString


        mov dh, 12
        mov dl, 25
        call setCursor
        lea dx, strRegressarPT3
        call printString

        call getMousePosRet

        ret
backspace:

        mov colunas, 0
        lea dx, paragrafo
        call printstring

        mov bl, linhas
        inc bl
        mov linhas, bl

        cmp linhas, 12
        je returns

        lea dx, paragrafo
        call printstring

        jmp inicio

    endp
```

←Limpa o ecrã.
←Limpa as variáveis.

←Lê um número do vetor de posições e escreve-o entre 2 traços verticais, fazendo lembrar uma secretária.

←Quando a cadeira está vazia, imprime o caractere com código ASCII 31 duas vezes, para assinalar que a cadeira não está ocupada.

←Permite o funcionamento do botão "Regressar" que direciona o utilizador para o menu.

←Quando estão 5 lugares preenchidos numa fila, passa para a fila seguinte.

FUNÇÃO "GUARDALOCAL"

```
;*****************************************************************
; guardarLocal - guarda a localizacao dos alunos num ficheiro
; input -
; output -
; destroys -
;*****************************************************************

    proc guardarLocal

        call setVideoMode

        call gerarNomeFile

        mov cx, 0
        lea dx, nomeFileACriar
        call fCreate

        mov handler0, ax

        call porinfonocop

        call setVideoMode

        mov dh, 4
        mov dl, 2
        call setCursor
        mov dx, offset strDisquete
        call printString

        mov dx, offset localizacao
        call printString

        mov dh, 10
        mov dl, 25
        call setCursor
        lea dx, strRegressarPT1
        call printString

        mov dh, 11
        mov dl, 25
        call setCursor
        lea dx, strRegressarPT2
        call printString

        mov dh, 12
        mov dl, 25
        call setCursor
        lea dx, strRegressarPT3
        call printString

        call getMousePosRet

        ret
    endp
```

←Limpa o ecrã.
←Através de inputs do utilizador cria o nome para o ficheiro binário do tipo *.cop.*

←Cria o ficheiro binário.

←Coloca o conteúdo do vetor de posições e o nome do ficheiro de texto ao qual se refere dentro do ficheiro binário.

←Permite o funcionamento do botão de "Regressar" direciona o utilizador para o menu.

FUNÇÃO "CONSULPOTINF"

```
;*************************************************************
; consultarPotInf-
; input -
; output -
; destroys -
;*************************************************************

    proc consultarpotinf

        call setvideomode

        call tratadocop

        lea si, stralunorecebido
        mov al, [si]
        mov charlido, al

        sub charlido, '0'

        mov ax, 10
        mov bl, charlido
        mul bl

        mov numalunocmp, al

        inc si
        mov al, [si]
        mov charlido, al

        sub charlido, '0'

        mov al, numalunocmp
        mov bl, charlido
        add al, bl

        mov numalunocmp, al

        mov indice, 0
        mov si, offset arraynums

cicloleitor:

        mov bl, [si]
        cmp bl, numalunocmp
        je  buscainfetados

        inc si

        mov al, indice
        inc al
        mov indice, al
        cmp al, 100
        je naohaaluno

        jmp cicloleitor

buscainfetados:
        call buscanumsafetados

        lea dx, filerecebidonocop
        call fOpen
        mov handler0, ax

GereNumeros:
        cmp nomesaimprimir, 0
        je foradaqui

        call tornanum

        lea di, arraydosafetados
        mov bl, intlido
        cmp bl, [di]
        je  QueroNome
        inc di

        cmp bl,[di]
        je  QueroNome
        inc di

        cmp bl,[di]
        je  QueroNome
        inc di

        cmp bl,[di]
        je  QueroNome

saltaCaracteresIrre:
        mov bx, handler0
        lea dx, charlido
        mov cx, 1
        call fRead

        cmp ax, 0
        je foradaqui

        cmp charlido, 10
        jne saltaCaracteresIrre

        jmp GereNumeros
```

←Limpa o ecrã.
←Coloca as primeiras 60 posições do ficheiro binário no vetor de posições e recebe a informação sobre o ficheiro de texto ao qual aquela organização de sala se refere.

←Coloca na variável o número de aluno introduzido pelo utilizador, em inteiros.

←Encontra a posição no vetor onde está o número pretendido.

←Chama uma função dedicada a encontrar os alunos que tenham cometido potenciais infrações.

←Percorre o ficheiro de texto em busca de encontrar o número e o nome dos alunos que tenham cometido potenciais infrações.

←Salta os caracteres irrelevantes.

```
QueroNome:
        lea dx, paragrafo
        call printstring

        lea dx, espacos
        call printString

        sub nomesaimprimir, 1
        mov dl, [di]
        call TornaChar
        mov dl, al
        call co
        mov dl, ah
        call co
        mov dl, 32
        xor dh, dh
        call co

        call SaltaSeparador

labelparaimprimironome:

        mov dl, charlido
        call co
        mov bx, handler0
        lea dx, charlido
        mov cx, 1
        call fRead
        cmp charlido, 10
        je GereNumeros

        jmp labelparaimprimironome

foradaqui:
        mov bx, handler0
        call fclose

        mov dh, 10
        mov dl, 25
        call setCursor
        lea dx, strRegressarPI1
        call printString

        mov dh, 11
        mov dl, 25
        call setCursor
        lea dx, strRegressarPI2
        call printString

        mov dh, 12
        mov dl, 25
        call setCursor
        lea dx, strRegressarPI3
        call printString

        call getMousePosRet

        ret

naohaaluno:
        mov dx, offset stralunonaoencontrado
        call printstring
        jmp foradaqui

    endp
```

←Ao encontrar o número dos alunos que podem ter copiado, este imprime o número desses alunos.

←Salta os espaços entre o número e o nome do aluno presentes no ficheiro de texto

←Imprime o nome dos alunos potenciais infratores.

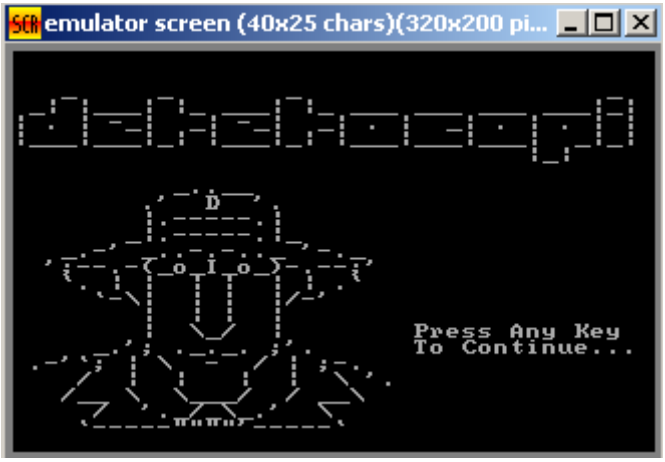
←Fecha o ficheiro e imprime um botão que permite voltar ao menu.

←Quando o número de aluno não existe, é impressa uma mensagem

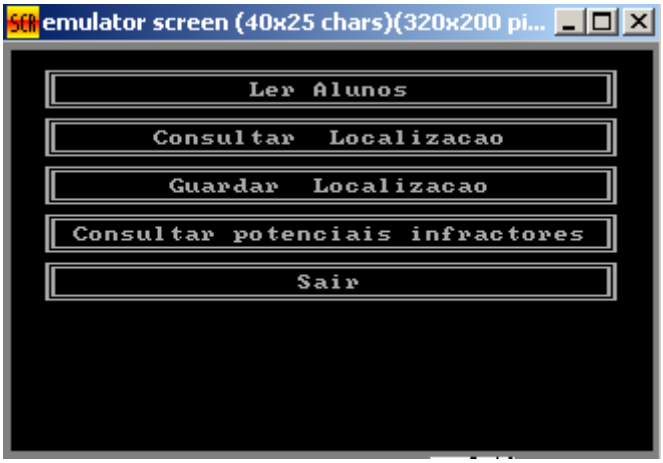
4. TESTES E RESULTADOS

Neste capítulo mostramos o programa desenvolvido e fazemos um *walkthrough* pelas variadas funcionalidades do programa.

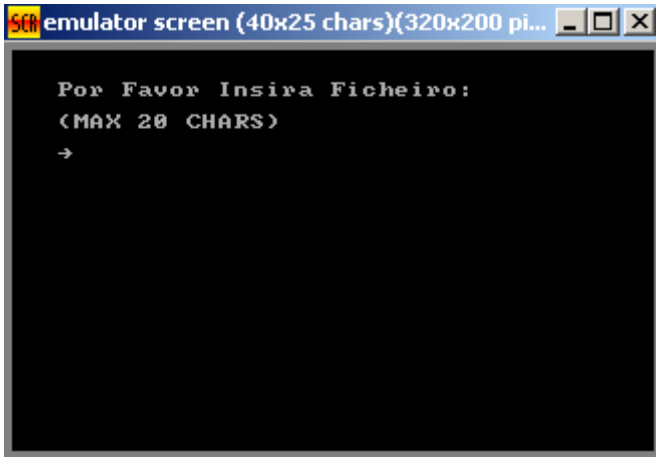
Ao abrir o programa, é nos apresentado um pequeno *loading screen* com uma breve apresentação.



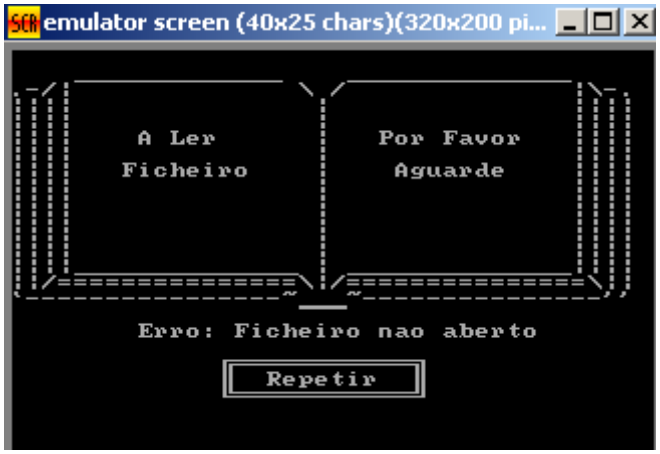
Depois de carregar num botão qualquer, o ecrã transaciona para o menu principal que contém as várias opções disponíveis, das quais o utilizador pode usufruir.



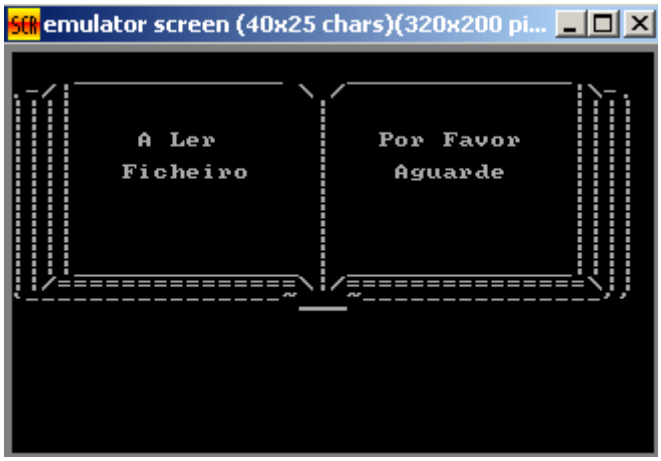
Ao escolher a opção “Ler Alunos”, somos redirecionados para um ecrã onde podemos escolher qual o ficheiro de texto que queremos ler.



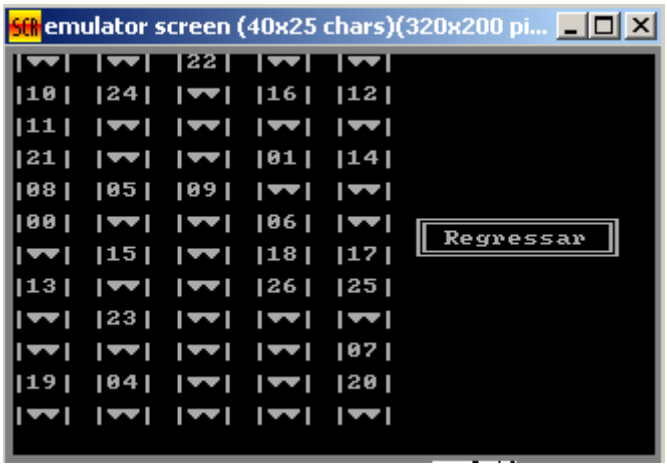
Se o ficheiro escolhido não estiver no sistema, o programa apresenta uma mensagem de erro e um botão para que seja possível repetir a leitura de um ficheiro.



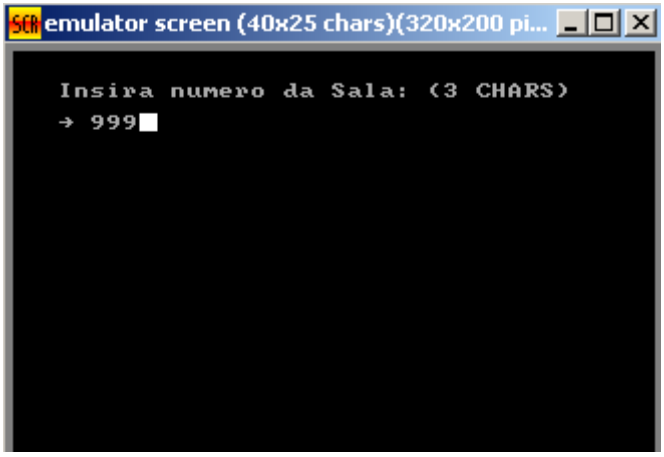
Se o ficheiro existir no sistema é nos mostrado um ecrã de transição enquanto o ficheiro é lido e, posteriormente, somos direcionados para o menu principal recebendo uma mensagem de que a leitura foi bem sucedida.



Ao escolher a opção de "Consultar Localização" é nos disposto um mapa de uma sala com 60 lugares, organizados em filas de 5 mesas, consoante a disposição gerada na função anterior.

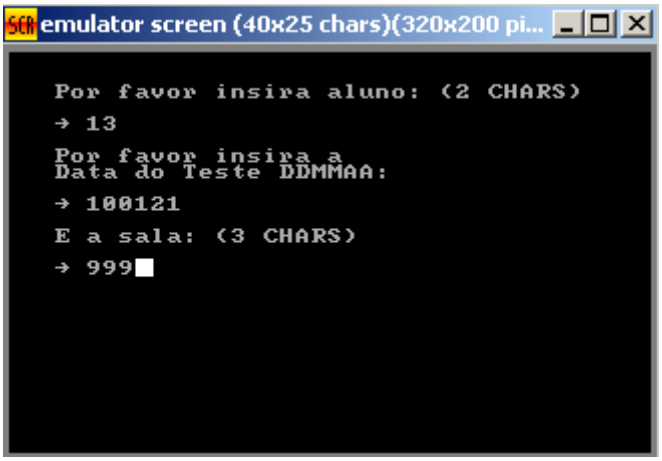


Ao escolher a função “Guardar Localização” somos direcionados para um ecrã onde nos é pedido o número da sala correspondente á disposição gerada.

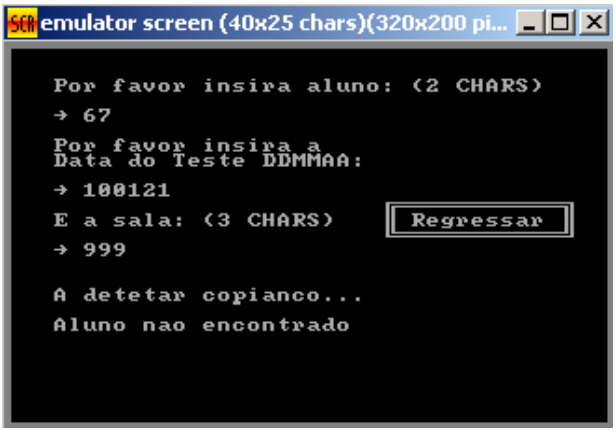


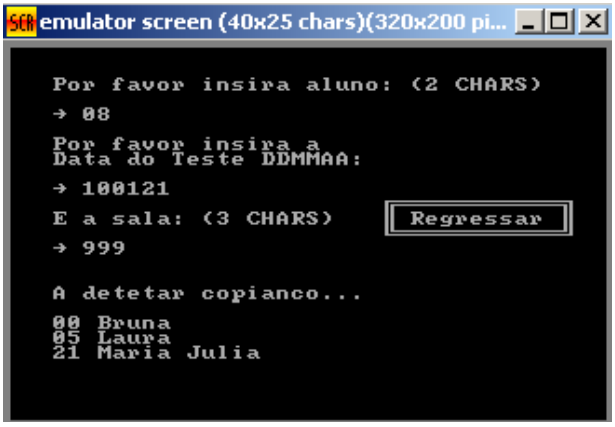
Após a informação ter sido guardada no ficheiro, recebemos uma mensagem de confirmação e é nos apresentado um botão para voltar para o menu.

Ao escolher a opção “Consultar Potenciais Infratores” somos levados para um ecrã onde é pedido ao utilizador o número do aluno identificado como infrator, o dia do teste e o número da sala onde estava o aluno.

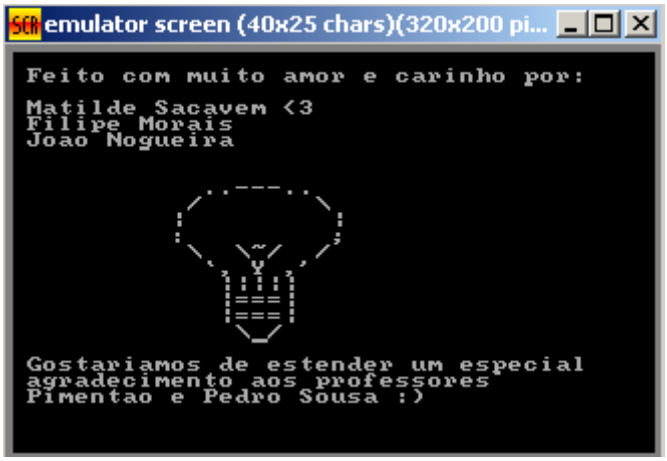


Se o número de aluno não existir recebemos uma mensagem a informar que o aluno não conta nas listas, se existir, imprime a lista dos alunos que o rodeiam ou avisa que não tinha ninguém perto dele durante o teste.





Ao selecionar a opção “Sair”, o utilizador é presenteado com um ecrã de despedida com uma mensagem de agradecimento e créditos aos criadores do programa.



5. CONCLUSÕES

Neste capítulo apresentamos as nossas reflexões científicas face ao trabalho realizado, e eventuais notas/comentários relevantes.

O nosso grupo concluiu que a implementação do trabalho final, de acordo com as especificações exigidas pelos docentes da Unidade Curricular foi um sucesso. Isto deve-se ao facto de o programa elaborado pelo grupo funcionar na totalidade e com relativamente elevada eficiência, como pudemos demonstrar ao correr e testar o programa. Ao longo deste trabalho surgiram alguns contratempos e obstáculos à progressão do mesmo, mas que com o nosso esforço e dedicação conseguimos ultrapassar.

6. REFERÊNCIAS BIBLIOGRÁFICAS

Materiais didáticos, na forma de slides, fornecidos pelos docentes João Paulo Pimentão e Pedro Alexandre da Costa Sousa, da Unidade Curricular de Sistemas Lógicos II.

Microprocessadores 20/21 → https://moodle.fct.unl.pt/course/view.php?id=6241