Licenciatura em Engenharia Informática

2020/2021

Relatório

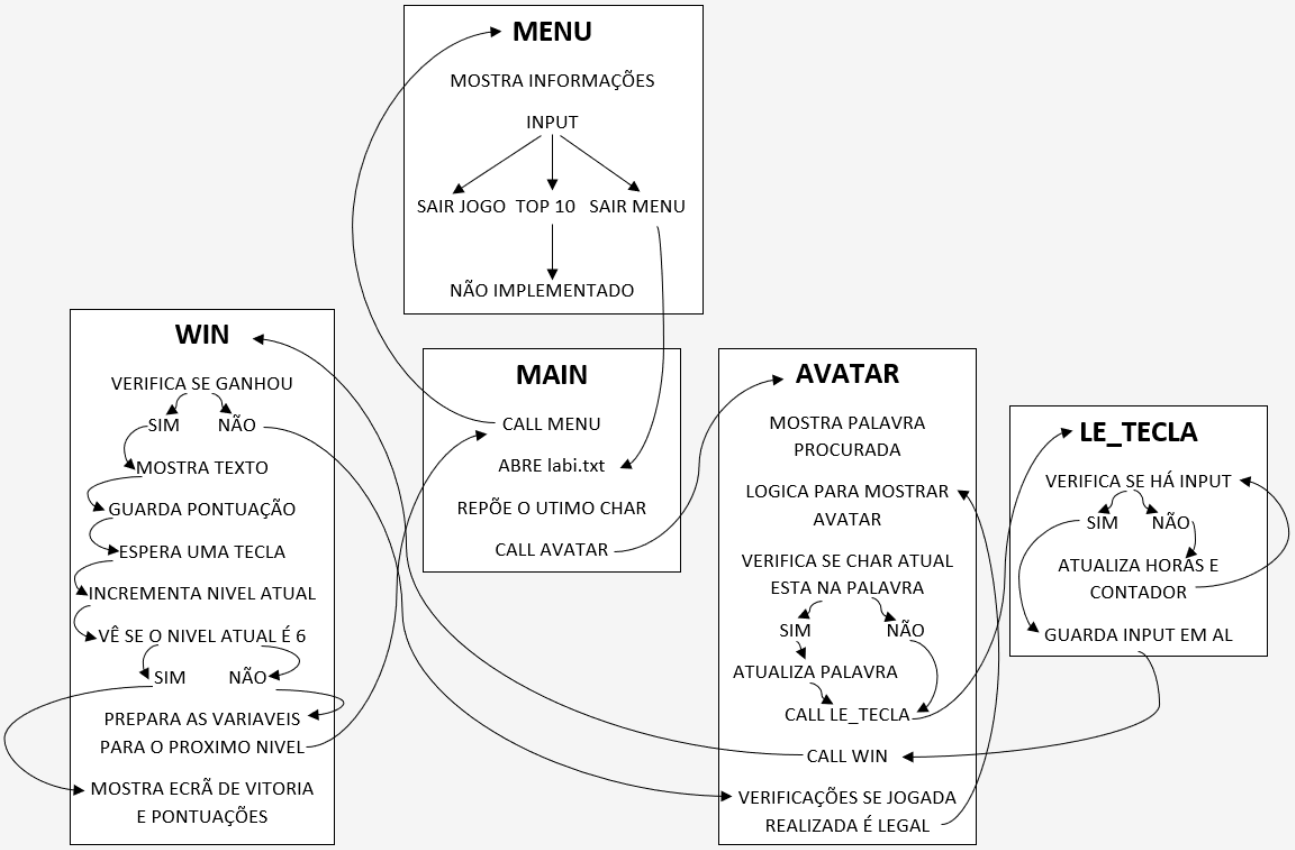
Trabalho Laboratorial de
Tecnologias e Arquiteturas de Computadores

Trabalho realizado por:
Gustavo Mateus - 2020138902
Hugo Gil - 2020130870

Objetivos

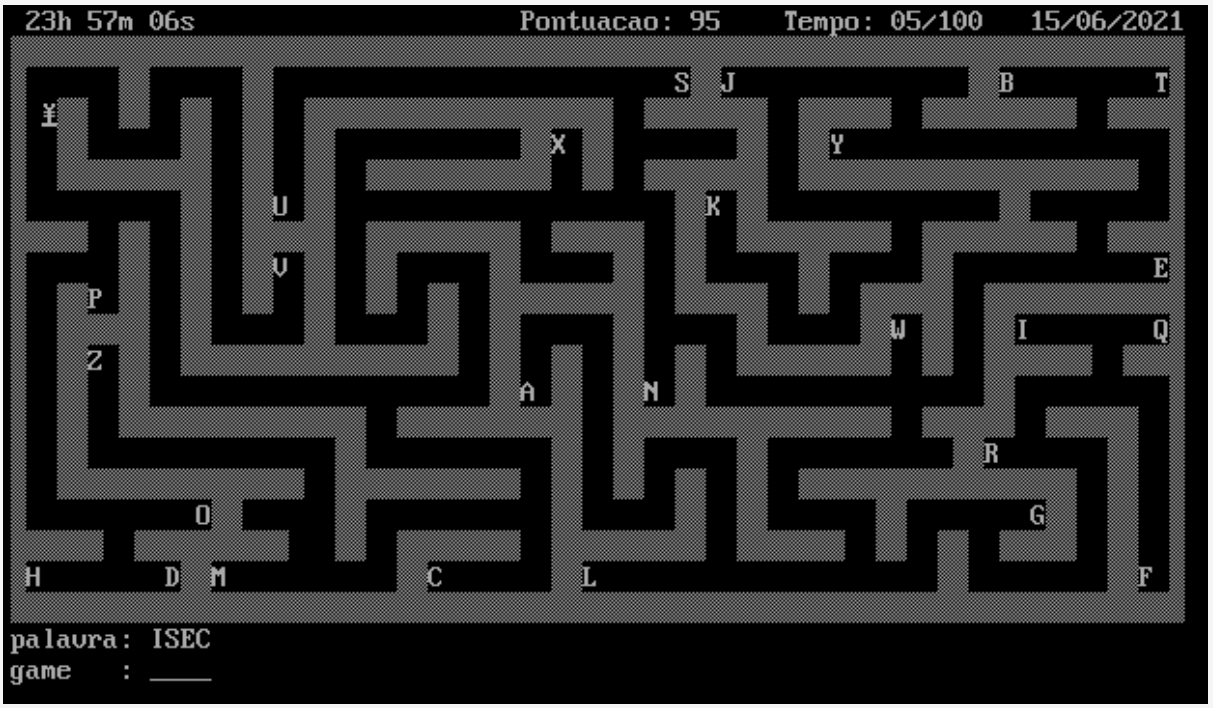
Com a realização deste trabalho pretendemos obter um conhecimento mais profundo e intuitivo das funcionalidades, benefícios e complicações de desenvolver um programa em assembly.
Ao depararmos-nos com problemas e explorarmos a linguagem à procura de maneiras de os resolver, conseguimos aprofundar os nossos conhecimentos de assembly e desenvolver uma apreciação maior por todas as ferramentas e linguagens que nos permitem trabalhar num nível de abstração mais alto.

Estrutura

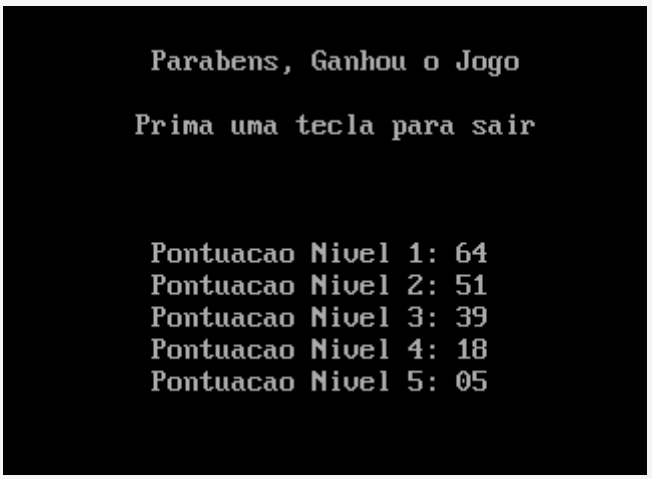

MENU
MOSTRA INFORMAÇÕES
INPUT
SAIR JOGO
TOP 10
SAIR MENU
NÃO IMPLEMENTADO
WIN
VERIFICA SE GANHOU
SIM
NÃO
MOSTRA TEXTO
GUARDA PONTUAÇÃO
ESPERA UMA TECLA
INCREMENTA NIVEL ATUAL
VÊ SE O NIVEL ATUAL É 6
SIM
NÃO
PREPARA AS VARIAVEIS PARA O PROXIMO NIVEL
MOSTRA ECRÃ DE VITORIA E PONTUAÇÕES
MAIN
CALL MENU
ABRE labi.txt
REPÕE O UTIMO CHAR
CALL AVATAR
AVATAR
MOSTRA PALAVRA PROCURADA
LOGICA PARA MOSTRAR AVATAR
VERIFICA SE CHAR ATUAL ESTA NA PALAVRA
SIM
NÃO
ATUALIZA PALAVRA
CALL LE_TECLA
CALL WIN
VERIFICAÇÕES SE JOGADA REALIZADA É LEGAL
LE_TECLA
VERIFICA SE HÁ INPUT
SIM
NÃO
ATUALIZA HORAS E CONTADOR
GUARDA INPUT EM AL

Funcionamento do programa

Nesta foto, encontra-se o menu, é aqui onde o programa começa e dá as 3 opções ao utilizador.
É para aqui também onde se vai quando o utilizador passa de nível.



Quando se seleciona a primeira opção, o jogo vai para este ecrã, que mostra o labirinto, a hora atual, a pontuação do nível atual, o tempo que o jogador tem para terminar o nível, a data, a palavra que se tem de obter e um sítio onde mostra as letras que vão sendo apanhadas.



Quando o jogador passa os níveis todos, vai para um ecrã que indica que ganhou o jogo e mostra a pontuação que obteve para cada nível.

Algoritmo subjacente a cada função desenvolvida

Funções já desenvolvidas e disponibilizadas nos ficheiros de apoio:

Não modificadas:

LER_TEMPO PROC -

Providenciada no ficheiro hms_dma.asm.
Obtém as horas do sistema e guarda-as em três variáveis separadas (Segundos, Minutos e Horas)

HOJE PROC -

Providenciada no ficheiro hms_dma.asm.
Obtém a data do sistema e coloca numa string na forma dd/mm/aaaa

GOTO_XY MACRO POSx,POSy-

Providenciada no ficheiro avatar.asm.
Coloca o cursor na posição x,y especificada.

MOSTRA MACRO STR -

Providenciada no ficheiro avatar.asm.
Coloca na posição atual do cursor a string determinada, desde que esta termine com o carácter “$”

APAGA_ECRAN PROC -

Providenciada no ficheiro avatar.asm
Esta função serve para apagar tudo o que estiver no ecrã.

IMP_FICH PROC -

Providenciada no ficheiro CriaFich.asm
Nós usamos esta função para imprimir no ecrã o labirinto, ou seja, a função importa o conteúdo dentro do ficheiro de texto labi.txt para o ecrã

Modificadas:

TRATA_HORAS PROC -

Providenciada no ficheiro hms_dma.asm.
Chama a função LER_TEMPO e verifica se os segundos se alteraram desde a última chamada, se sim passa as horas/minutos/segundos para uma string formatada corretamente e mostra a mesma no ecrã.
Chama também a função horas.

Modificações:
Contém a lógica do contador, incrementado-o a cada segundo e passando para uma string de maneira a mostrar o mesmo no ecrã.
Contém também a lógica dos pontos, tratando-os de uma maneira semelhante ao contador mas decremento-os ao invés de os incrementar.
A função VERIFICA_DERROTA é chamada para verificar se o tempo terminou.

LE_TECLA -

Providenciada no ficheiro avatar.asm
Lê a tecla e guarda-a em AL.

Modificações:
Nesta função chama-se o TRATA_HORAS continuamente enquanto não for pressionada uma tecla, de maneira que o tempo e contador atualizem todos os segundos.

MAIN PROC -

Função já desenvolvida e providenciada no ficheiro avatar.asm
Possui a lógica principal do jogo, chamando o apagar_ecrã, menu e avatar.

Modificações:
Adicionamos código para permitir a reposição do último caractere que o avatar pisou antes de avançar de nível, corrigindo assim um bug que explicamos em maior detalhe nas principais opções de desenvolvimento.

AVATAR PROC -

Inicia-se o nível e é imprimida a palavra procurada no nível atual.
Guarda-se na variável Car o caráter que estiver debaixo do avatar.
É verificado se o caráter está presente na palavra do nível atual.
Se sim, é obtido esse caráter e continua-se a verificar a palavra até ao seu fim caso haja repetições.
Quando se chegar ao final da palavra, o programa continua e é imprimido o avatar.
Chama a função LE_TECLA descrita anteriormente.
Chama-se também de seguida a função WIN para verificar se já foram coletadas as letras todas.
Ao receber o input, de seguida é verificado que seta foi pressionada e o avatar avança para a direção respetiva.
Depois de avançar é lido o caractere sob o qual o avatar pisa e se este for uma parede (com código ASCII 177) o movimento que foi feito para o levar a essa posição é revertido. Isto tudo toma lugar rapidamente, não sendo percetível ao jogador.

Funções desenvolvidas de raiz:

MENU PROC -

Nesta função é dado ao utilizador 3 opções, jogar, ver o top 10 , que não foi implementado e sair do programa.
Quando o utilizador escolhe jogar, o programa inicia o jogo ao chamar a função IMP_FICH, que imprime o labirinto e depois chama o AVATAR que inicia o nível.
Cada nível que é passado, é chamado o MENU e incrementa-se a variável Nivel para mostrar o número do próximo nível.

WIN PROC-

- Verifica se ainda falta alguma letra para completar a palavra procurada, se faltar alguma o programa continua.
Se já tiver todas as letras, o programa vê qual é o próximo nível e preenche as variáveis de acordo com o número do nível.
Mostra a pontuação que o jogador obteve no nível que passou e guarda a mesma na variável indicada para o nível que foi vencido.
Se o próximo nível for o 6, quer dizer que o jogador passou os níveis todos e indica ao jogador que ele ganhou.

VERIFICA_DERROTA PROC -

- A função verifica se o contador já chegou ao seu limite.
Se sim, o programa indica que o utilizador perdeu e pergunta se quer tentar de novo desde o início. Caso o jogador pretenda recomeçar, as variáveis são todas preenchidas com os valores do primeiro nível.

Principais opções de desenvolvimento

Durante o desenvolvimento do jogo um problema com que nos deparamos foi permitir a atualização constante das horas e do cronômetro, sem comprometer a resposta imediata a inputs do jogador.
Para foi necessário evitar prender o programa enquanto este espera por o input do jogador.
Ficou também rapidamente claro que o programa esperava pelo input na função LE_TECLA de uma maneira que não permitia que nenhum bloco de código corresse simultaneamente.
Para resolver essa situação implementamos as seguintes linhas de código:

```
sem_tecla:
    call Trata_Horas
    MOV AH, 0BH
    INT 21h
    CMP AL,0
    JE sem_tecla

    mov     ah,08h
    int     21h      ;le a tecla q pressionas
```

Utilizando o INT 21h / AH=0Bh obtemos o status do input no registo AL, sendo AL = 00h se nenhum caractere estiver disponível e AL = 0FFh se houver um disponível.
De seguida o CMP permite ao programa chamar continuamente o Trata_Horas, e apenas avançar para o INT 21h / AH=08h depois do jogador premir uma tecla.

Além disso, durante a implementação dos próximos níveis, quando se mostrava o labirinto estávamos com um problema em que a última letra coletada desaparecia, então para resolver o problema usamos o seguinte código:

```
ultimo_caracter: ;Repoe ultimo caracter que o avatar pisou antes de passar para o proximo nivel
    goto_xy POSy, POSx
    mov     ah, 02h
    mov     dl, Car
    int     21H
    jmp voltar_avatar
```

Aqui com o POSy e POSx, nós conseguimos ir para a última localização onde o avatar estava quando acabou o jogo e imprimir lá o Car que irá ser a última letra obtida.
Com esta solução foi causado outro problema em que a última letra aparece também na posição inicial do próximo nível.
Conseguimos resolver o problema em causa com este código:

```
goto_xy 3, 3
mov     ah, 02h
mov     dl, " "          ; Repoe Caracter guardado
int     21H
```