Laboratório de Computadores | 2019/2020 | MIEIC

Relatório do Projeto de LCOM

Diogo André Barbosa Nunes | up201808546

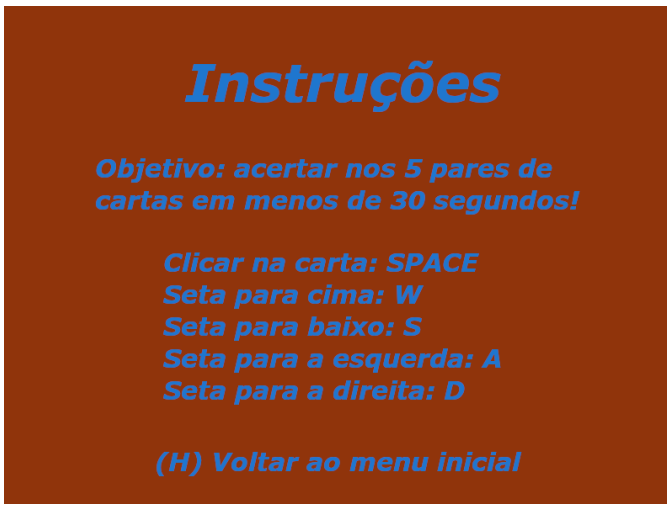
João Pedro Olinto Dossena | up201800174

1. Instruções de utilização do programa

Ecrã inicial

Ao iniciar o programa, é apresentado o menu inicial, que possui o título do jogo, a unidade curricular, o ano letivo, os nomes dos realizadores do jogo e as opções de Jogar (clicando na tecla J), ver as Instruções (tecla I) ou Sair do programa (tecla ESC).

Menu de instruções



Este menu apenas apresenta os botões que terão que ser utilizados para jogar, apenas dando para voltar para o menu inicial, clicando na tecla H (de Home).

Jogar

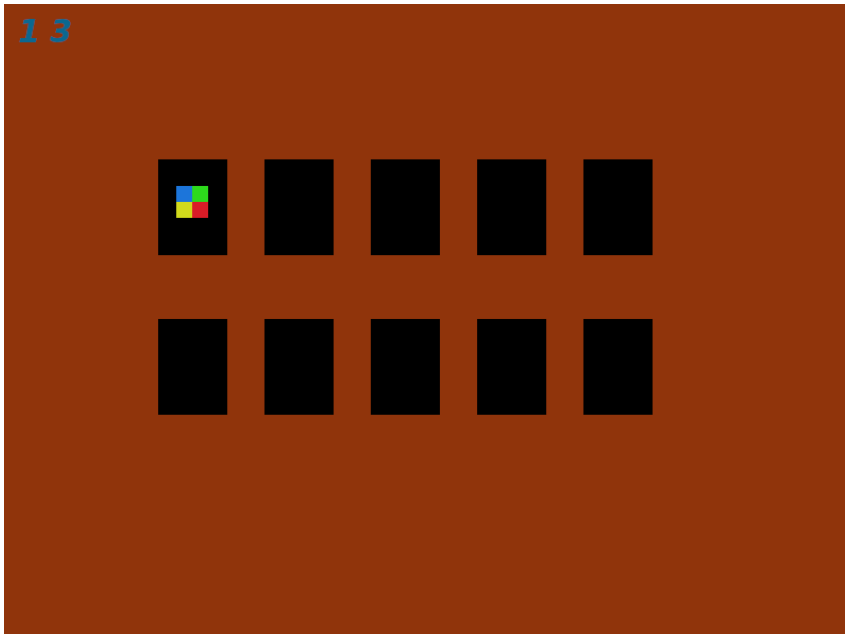


Imagem do jogo aos 13 segundos, sem ter descoberto nenhum par e na posição inicial.

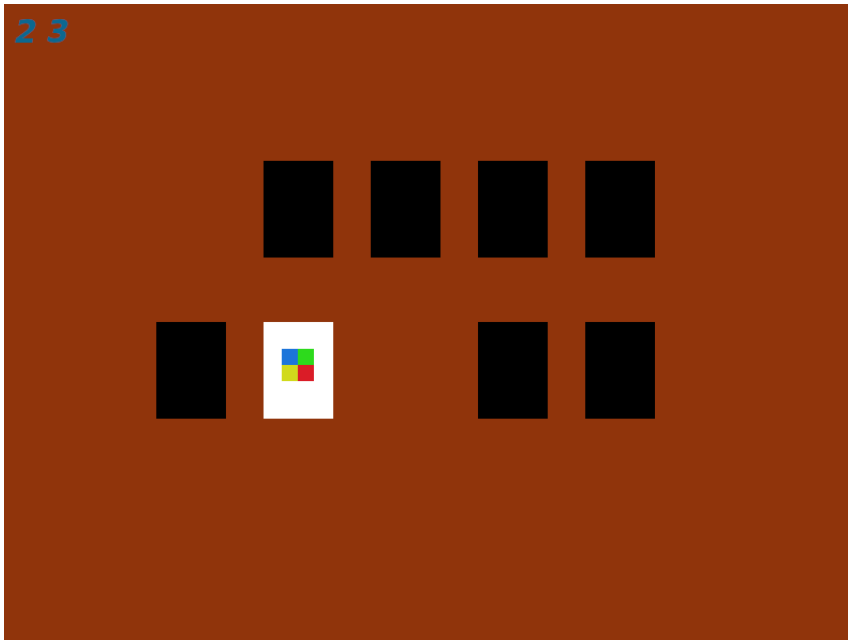
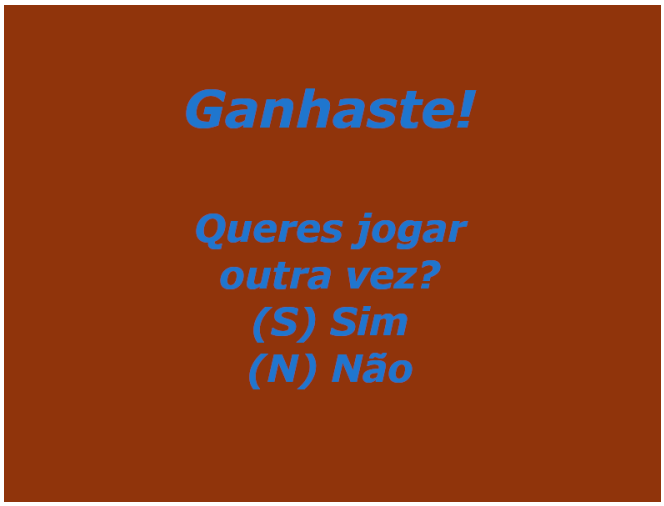


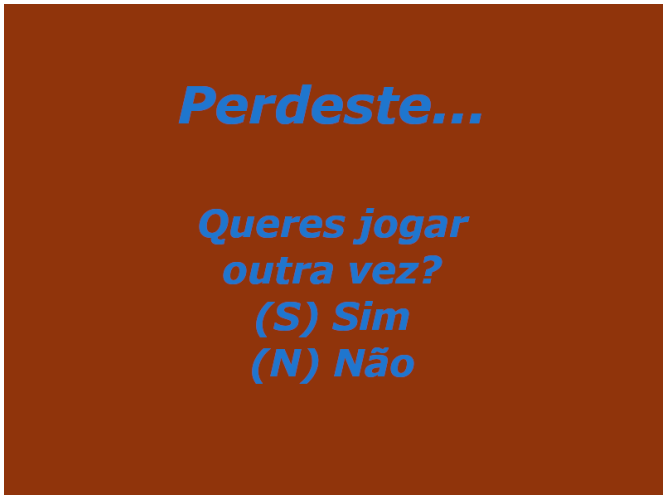
Imagem do jogo aos 23 segundos, com 1 par já descoberto e na posição da carta branca virada para cima.

O jogo consiste em desafiar o utilizador a encontrar os 5 pares de cores em menos de 30 segundos.

Caso consiga:



Caso não consiga:



Em ambos os casos, se clicarmos na tecla S, voltamos a jogar, e se clicarmos na tecla N, voltamos ao menu inicial.

2. Estado do projeto

Dispositivo	Utilização	Interrupção
Timer	Contador do tempo-limite do jogo	Sim
Teclado	Interface com o jogo, mudança de menus e atualização do ecrã	Sim
Placa Gráfica	Desenho de imagens	Não

Timer

Este dispositivo é usado no nosso programa para contar o tempo desde o início do jogo. Com a ajuda das interrupções, quando passa 1 segundo, o contador é incrementado e desenhado no ecrã. Se o utilizador acabar o jogo antes do contador chegar aos 30 segundos, ganha, caso contrário, perde.

Teclado

A função do teclado é gerir o programa em si. O jogo e a mudança de menus é apenas feita através do teclado (como já foi dito nas instruções de utilização) e o ecrã atualiza sempre que seja clicada uma qualquer tecla.

Placa Gráfica

Este dispositivo desenha todas as imagens visíveis no ecrã (no formato xpm e no modo RGB (8:8:8)). O modo de vídeo que usamos é 0x115, de resolução 800 por 600 pixéis.

3. Organização e Estrutura do Código

xpms

Este módulo inclui a maioria das funções de desenho do programa, incluindo menus, cartas, letras e números.

game

Este módulo é onde é gerido todo o jogo (*void interrupt_handler(Game *game, Game_state *states)*), desde o seu início (*Game *start_game()*) até ao seu fim (*void end_game(Game *game)*).

Possui ainda 2 funções fundamentais ao desenvolvimento do programa e da máquina de estados implementada para controlar o jogo:

- *int detect_keyboard_event(Game *game, Game_state *states);*
- *int manage_states(Game *game, Game_state *states)*.

cards

Este módulo possui a declaração da estrutura de cada carta todas as funções que envolvam cartas:

- *int create_deck(card_t deck[NUM_CARDS])*: esta função prepara o deck de um jogo, chamando as funções de escolha de cores (*void shuffleColors(uint8_t allColors[NUM_CLRS]*) e de aleatoriedade de posições (*void shuffleDeck(card_t deck[NUM_CARDS]*);
- *int drawCards(card_t deck[NUM_CARDS])*: função que desenha as cartas, dependendo se estão viradas para cima, para baixo ou se já foram encontradas;
- *int turnCard(card_t deck[NUM_CARDS])*: apenas torna uma carta virada para cima caso esteja virada para baixo e vice-versa;
- *int analise_pairing(card_t deck[NUM_CARDS])*: analisa se o par de cartas viradas para cima são iguais, de modo a completar um par de cartas descoberto ou não.

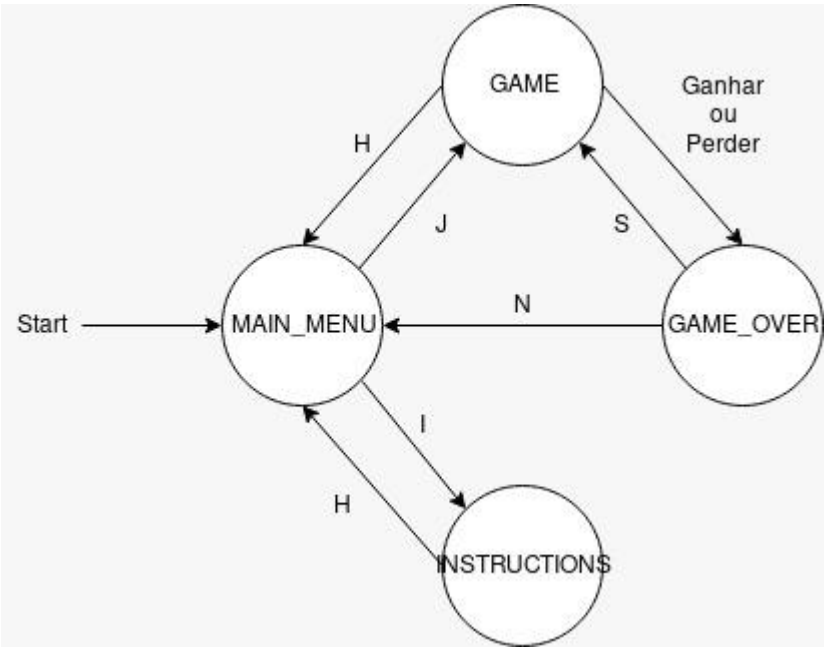
codes, i8042, i8254, ps2 e vbe

Estes *headers* contêm apenas variáveis que foram definidas para serem usadas mais facilmente noutros ficheiros.

State_machine

Este módulo reserva memória para o jogo no início do programa (*Game_state *new_states()*) e liberta-a quando termina (*void delete_states(Game_states *game_states*).

Também é neste *header* que é declarada a nossa máquina de estados, que é representada pelo seguinte gráfico:



dot

Este módulo contém a declaração da estrutura do apontador usado para as cartas durante o jogo, e 4 funções de movimento, conforme se queira deslocar para cima, para baixo, para a esquerda ou para a direita (*move_up(dot *ponto), move_down(dot *ponto), move_left(dot *ponto), move_right(dot *ponto)*, respetivamente).

timer e utils

Estes módulos foram também usados no laboratório 2, e possuem, entre outras, funções para subscrever e cancelar as interrupções do timer (*int timer_subscribe_int(uint8_t *bit_no) e int timer_unsubscribe_int()*).

keyboard

Este módulo foi, tal como o timer, utilizado no laboratório 3 das aulas práticas, e também possui funções para o mesmo efeito (*int keyboard_subscribe_int(uint8_t *bit_no) e int keyboard_unsubscribe_int()*.).

video

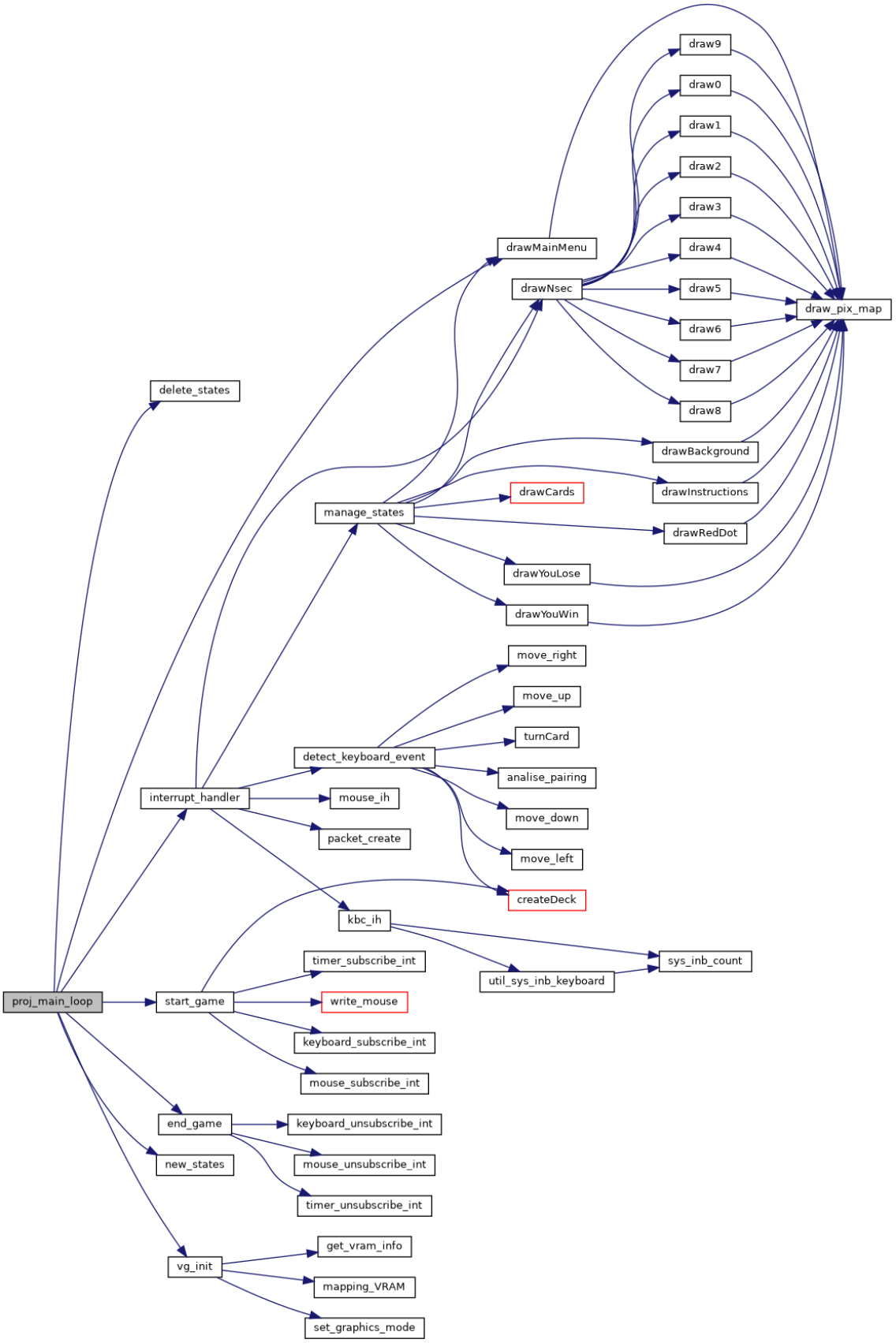
Este módulo foi utilizado no laboratório 5 e possui funções para inicializar o modo de vídeo na resolução desejada (*void* vg_init(uint16_t mode)*, que chama as funções *int get_vram_info(uint16_t mode), int mapping_VRAM() e int set_graphics_mode(uint16_t mode)*), e a função principal de desenho, isto é, a função chamada para qualquer imagem que se queira desenhar no ecrã (*int draw_pix_map(xpm_image_t img, uint16_t x, uint16_t y*).

proj

Este é o módulo principal, porque é a base para todos os restantes módulos.

Inicializa o modo de vídeo (*vg_init(MODE_800x600)*), cria a estrutura do jogo e da máquina de estados correspondente (*start_game() e new_states()*), monitoriza todo o jogo dentro da função *interrupt_handler(game, states)*, e acaba o jogo cancelando todas as interrupções subscritas no início do jogo e liberta a memória utilizada pela máquina de estados (*end_game(game) e delete_states(states)*).

4. Call Graph


draw9
draw0
draw1
draw2
draw3
draw4
draw5
draw6
draw7
draw8
drawMainMenu
drawNsec
draw_pix_map
delete_states
drawBackground
drawInstructions
drawCards
manage_states
drawRedDot
drawYouLose
drawYouWin
move_right
move_up
turnCard
detect_keyboard_event
analise_pairing
interrupt_handler
mouse_ih
move_down
packet_create
move_left
createDeck
kbc_ih
sys_inb_count
timer_subscribe_int
util_sys_inb_keyboard
proj_main_loop
start_game
write_mouse
keyboard_subscribe_int
mouse_subscribe_int
end_game
keyboard_unsubscribe_int
new_states
mouse_unsubscribe_int
timer_unsubscribe_int
get_vram_info
vg_init
mapping_VRAM
set_graphics_mode

5. Conclusão

Desde o início do semestre que esta UC foi a mais desafiante de todo o curso, para ambos.

Ambos concordamos que as aulas teóricas ajudaram bastante a complementar os laboratórios realizados nas aulas práticas, mas achamos que em vez de aulas tão longas, era preferível dividir, tanto as teóricas como as práticas, em 2 ou 3 aulas durante toda a semana, de forma a poder tirar dúvidas mais do que 1 vez por semana presencialmente.

Também foi um pouco desvantajoso o facto de, por vezes, estar nas aulas práticas a terminar um laboratório sobre um certo dispositivo e nas aulas teóricas já estarmos a conhecer o dispositivo seguinte. Penso que isto podia ser facilmente resolvido com as aulas distribuídas durante a semana, como disse acima.

Para além de sentirmos a necessidade de recorrer ao apoio dos nossos professores das aulas práticas constantemente, também achamos que os guiões dos laboratórios, que estão muito bem organizados e dizem rigorosamente tudo o que temos que implementar, também deveriam explicitar de alguma forma como implementar o código pedido. Não estou a dizer que nos deviam dar o código feito, mas, por exemplo, que um dos dispositivos dados fosse dado apenas como exemplo e que nos fosse mostrada a estrutura pedida.

De resto, penso que foi a primeira UC que nos deu realmente muito trabalho, tanto em termos de tempo, como em termos de implementação de código.

Resumindo, ambos sentimos dificuldades, mas ambos nos esforçamos de igual forma para realizar este projeto.