```
;Diogo Martins Alves N° 86980
;Xavier Abreu Dias N° 87136

; Declaração de constantes
;Stack pointer
SP_INIC EQU FDFFh
;Interrupções
TAB_INT0 EQU FE00h
TAB_INT1 EQU FE01h
TAB_INT15 EQU FE0Fh

MASCARA_INT EQU FFFAh
;I/O
IO_STAT EQU FFFDh
IO_READ EQU FFFFh
IO_CURSOR EQU FFFCh
IO_WRITE EQU FFFEh
LCD_CONTROL EQU FFF4h
LCD_WRITE EQU FFF5h
TIMER_COUNT EQU FFF6h
TIMER_START EQU FFF7h
DISPLAY_0 EQU FFF0h
DISPLAY_1 EQU FFF1h
DISPLAY_2 EQU FFF2h
DISPLAY_3 EQU FFF3h
LEDS EQU FFF8h

LIMPAR_JANELA EQU FFFFh
FIM_TEXTO EQU '@'
;cores
BRANCO EQU 0000h
PRETO EQU 0001h
ENCARNADO EQU 0002h
VERDE EQU 0003h
AZUL EQU 0004h
AMARELO EQU 0005h
VAZIO EQU 0006h

;Definição das variáveis
         ORIG 8000h
RandNum WORD 0000h  ; Valor aleatório do algoritmo aleatório
MascaraAlgoritmoAleatorio WORD 9c16h    ; Mascara do algoritmo aleatório
ContagemTempo WORD 0000h    ;Contagem decrescente do tempo de cada tentativa
TempoTotalJogada WORD 0000h     ;Tempo total que um jogador demorou a terminar a sua jogada
Sequencia TAB 6 ; reserva espaço para a sequência de cores
Sequencia_Copia TAB 6
TentAtual TAB 6 ; reserva espaço para a tentativa atual do jogador
TentAtual_Copia TAB 6
Avaliacao TAB 6 ; Avaliação da jogada atual
Desistir WORD 0000h
IniciarNovoJogo WORD 0000h
NumJogadores WORD 0001h
NumeroCoresSequencia WORD 0004h
TempoJogo WORD 0078h ;tempo de jogo em segundos escolhido pelo utilizador no inicio do
programa
JogadorAtual WORD 0000h
MelhorJogadasAcerto WORD 000fh
MelhorJogador WORD 0000h
;Estatisticas de cada jogo
MelhorTempoJogada WORD 0fffh
AccTemposJogo WORD 0000h
AccJogadasAcerto WORD 0000h
NumVitorias WORD 0000h
;Estatisticas dos jogadores
MelhorTempoJogo_Jogador STR 0fffh,0fffh,0fffh,0fffh,0fffh,0fffh,0fffh,0fffh,0fffh
AccTemposJogo_Jogador TAB 9
AccJogadasAcerto_Jogador TAB 9
NumVitorias_Jogador TAB 9

VarTexto1 STR '================-MASTERMIND-================',FIM_TEXTO
VarTextoM STR '                  JOGADOR X                 ',FIM_TEXTO
```

```
VarTexto2 STR '-----------------------------------------',FIM_TEXTO
VarTexto3 STR '   Tentativas  |          Jogadas          |',FIM_TEXTO
VarTexto4 STR '  º Tentativa  |                           |',FIM_TEXTO
VarTexto5 STR '     Solução   |                           |',FIM_TEXTO
VarTexto6 STR '|_____________________________________________________________|',FIM_TEXTO
VarTexto7 STR '|                        MASTERMIND                           |',FIM_TEXTO
VarTexto8 STR '|_____________________________________________________________|',FIM_TEXTO
VarTexto9 STR 'Neste jogo, o computador irá gerar uma sequência aleatória de ',FIM_TEXTO
VarTextoA STR 'cores, de entre seis (B: Branco, P: Preto, E: Encarnado, V:   ',FIM_TEXTO
VarTextoB STR 'Verde, Z: Azul, A: Amarelo) e o jogador terá 10 tentativas pa-',FIM_TEXTO
VarTextoC STR 'ra tentar acertar nessa sequência. Em cada tentativa, o jogador',FIM_TEXTO
VarTextoD STR 'dispõe de 1,2 ou 4 minutos e cada tentativa será avaliada  com',FIM_TEXTO
VarTextoE STR 'uma sequência de "P" e "B", sendo que cada "P" significa  que ',FIM_TEXTO
VarTextoF STR 'o jogador acertou na cor e posição de uma peça e cada "B" signi-',FIM_TEXTO
VarTextoG STR 'fica que o jogador acertou apenas na cor de uma peça.    ',FIM_TEXTO
VarTextoH STR 'No caso de querer desistir, basta clicar no Botão de pressão 0.',FIM_TEXTO
VarTextoI STR 'Carregue numa tecla para iniciar o jogo...',FIM_TEXTO
VarTextoJ STR 'PERDEU O JOGO!',FIM_TEXTO
VarTextoK STR 'GANHOU O JOGO!',FIM_TEXTO
VarTextoL STR 'Insira o número de jogadores: ',FIM_TEXTO
VarTextoN STR 'Insira o número de cores da sequência (4, 5 ou 6): ',FIM_TEXTO
VarTextoO STR 'Insira o tempo de jogo, em minutos (1, 2 ou 4): ',FIM_TEXTO
VarTextoP STR 'Prima uma tecla para passar a jogada ao próximo jogador...',FIM_TEXTO
VarTextoQ STR 'Prima o botão 1 para iniciar um novo jogo...',FIM_TEXTO
VarTextoR STR 'Vencedor: Jogador ',FIM_TEXTO
VarTextoS STR 'Nenhum ',FIM_TEXTO
VarTextoT STR 'Melhor Tempo: ',FIM_TEXTO
VarTextoU STR 'Tempo Médio: ',FIM_TEXTO
VarTextoV STR 'Média Jogadas para acerto: ', FIM_TEXTO
VarTextoX STR 'Estatísticas do Jogador', FIM_TEXTO
VarTextoY STR 'Jogador sem vitórias',FIM_TEXTO
VarTextoZ STR 'Jogo sem vitórias',FIM_TEXTO


;Codigo
ORIG    0000h
JMP     Inicio

;Rotina de interrupção 0 (botão 0) - Quando clicado, faz o jogador terminar o jogo atual com
o resultado derrota
RotinaInt0: PUSH R1
            CALL MostrarSol
            MOV R1, 1
            MOV M[Desistir],R1
            POP R1
            RTI

;Rotina de interrupção 1 (botão 1) - Quando clicado no final de um jogo, é iniciado um novo
jogo
RotinaInt1: PUSH R1
            MOV R1, 1
            MOV M[IniciarNovoJogo], R1
            POP R1
            RTI

;Rotina de interrupção 15 (temporizador)
RotinaInt15: PUSH R1
             INC M[TempoTotalJogada]
             DEC M[ContagemTempo]
             BR.Z FimTempo
             CALL EscreverDisp7Seg
             MOV R1, 10
             MOV M[TIMER_COUNT], R1; 10*100ms = 1s
             MOV R1, 1
             MOV M[TIMER_START], R1
             POP R1
             RTI
    FimTempo:MOV R1, 1
             MOV M[Desistir],R1 ;Quando o tempo acaba, o efeito é o mesmo do que quando o
             jogador desiste
             POP R1
```

```
            RTI


;Rotina que mete todos os digitos do display de 7 segmentos a zeros
LimparDisp7Seg:     MOV M[DISPLAY_0], R0
                    MOV M[DISPLAY_1], R0
                    MOV M[DISPLAY_2], R0
                    MOV M[DISPLAY_3], R0
                    RET

;Rotina para limpar a janela de texto
LimparJanela:   PUSH R2
                MOV R2, LIMPAR_JANELA
                MOV M[IO_CURSOR], R2
                POP R2
                RET

;Rotina para limpar o LCD
LimparLCD:  PUSH R1
            MOV R1, 8020h
            MOV M[LCD_CONTROL], R1
            POP R1
            RET

;Rotina que mostra o ecra inicial e gera a semente para o algoritmo
EcraInicial:PUSH R1
            PUSH R2
            PUSH VarTexto6
            PUSH 0000h
            CALL EscString
            PUSH VarTexto7
            PUSH 0100h
            CALL EscString
            PUSH VarTexto8
            PUSH 0200h
            CALL EscString
            PUSH VarTexto9
            PUSH 0300h
            CALL EscString
            PUSH VarTextoA
            PUSH 0400h
            CALL EscString
            PUSH VarTextoB
            PUSH 0500h
            CALL EscString
            PUSH VarTextoC
            PUSH 0600h
            CALL EscString
            PUSH VarTextoD
            PUSH 0700h
            CALL EscString
            PUSH VarTextoE
            PUSH 0800h
            CALL EscString
            PUSH VarTextoF
            PUSH 0900h
            CALL EscString
            PUSH VarTextoG
            PUSH 0a00h
            CALL EscString
            PUSH VarTextoH
            PUSH 0b00h
            CALL EscString
            PUSH VarTextoI
            PUSH 0c00h
            CALL EscString
            MOV R2, R0
EsperaSeed: MOV R1, M[IO_STAT]
            INC R2
            CMP R1, R0
            BR.Z EsperaSeed ;Verifica se foi premida alguma tecla
```

```
            MOV M[RandNum], R2
            MOV M[IO_STAT], R0
            MOV M[IO_READ], R0 ;limpa o valor introduzido no teclado
            POP R2
            POP R1
            RET


; Rotina que mostra o segundo ecra
    SegundoEcra:PUSH R1
                PUSH R2
                PUSH VarTextoL
                PUSH 0000h
                CALL EscString
LoopNumPlayers: MOV R1, M[IO_STAT] ;Verifica se foi premida alguma tecla
                CMP R1, R0
                BR.Z LoopNumPlayers
                MOV R1, M[IO_READ] ;R1 fica com o código do caracter da tecla premida
                CMP R1, '1'     ;NumJogadores tem que estar entre 1 e 9
                BR.N LoopNumPlayers
                CMP R1, '9'
                BR.P LoopNumPlayers
                MOV R2, 001eh
                MOV M[IO_CURSOR], R2
                CALL EscCar
                SUB R1, '0'
                MOV M[NumJogadores], R1

                PUSH VarTextoN
                PUSH 0100h
                CALL EscString
LoopNumCores:   MOV R1, M[IO_STAT]      ;Verifica se foi premida alguma tecla
                CMP R1, R0
                BR.Z LoopNumCores
                MOV R1, M[IO_READ] ;R1 fica com o caracter da tecla premida
                CMP R1, '4'         ;NumeroCoresSequencia tem que estar entre 4 e 6
                BR.N LoopNumCores
                CMP R1, '6'
                BR.P LoopNumCores
                MOV R2, 0133h
                MOV M[IO_CURSOR], R2
                CALL EscCar
                SUB R1, '0'
                MOV M[NumeroCoresSequencia], R1

                PUSH VarTextoO
                PUSH 0200h
                CALL EscString
LoopValorTempo: MOV R1, M[IO_STAT]  ;Verifica se foi premida alguma tecla
                CMP R1, R0
                BR.Z LoopValorTempo
                MOV R1, M[IO_READ] ;R1 fica com o caracter da tecla premida
                CMP R1, '1'     ;TempoJogo tem que estar entre 1 e 4, mas não pode ser 3
                BR.N LoopValorTempo
                CMP R1, '4'
                BR.P LoopValorTempo
                CMP R1, '3'
                BR.Z LoopValorTempo
                MOV R2, 0230h
                MOV M[IO_CURSOR], R2
                CALL EscCar
                SUB R1, '0'
                MOV R2, 60
                MUL R1, R2
                MOV M[TempoJogo], R2
                CALL DelayMaior
                POP R2
                POP R1
                RET

;Rotina para ler uma tentativa de cores inserida pelo jogador (recebe o numero da tentativa
```

```
atual pelo stack)
EsperarTentativa:   PUSH R1
                    PUSH R2
                    PUSH R3
                    PUSH R4
                    PUSH R5
                    MOV R2, M[SP+7] ;número da tentativa atual
                    MOV R5, 0100h
                    MUL R2, R5
                    MOV R1, M[NumeroCoresSequencia]
                    MOV R4, 6       ;Número de cores maximo
                    SUB R4, R1
                    MOV R1, 2       ;Numero de espaços entre as cores/2
                    MUL R1, R4
                    ADD R5, R4
                    ADD R5, 0413h           ;R5 fica com a posição de onde será escrita a
                    primeira cor da tentativa em questão
                    MOV R2, R0
        Espera:     CMP M[Desistir], R0     ;Caso o jogador desista ou o tempo acabe, a
        rotina salta para o fim
                    JMP.NZ Fim_EsperarTentativa
                    MOV R1, M[IO_STAT]
                    CMP R1, R0
                    BR.Z Espera ;Verifica se foi premida alguma tecla
                    MOV R3, R2
                    ADD R3, TentAtual ;R3 fica com o endreço de onde será guardada a cor
                    atual
                    MOV R1, M[IO_READ] ;R1 fica com o caracter da tecla premida
                    CMP R1, 'b'         ;É verificado se a tecla premida foi alguma das
                    permitidas e, caso seja, é guardada em memória e impressa no ecra
                    BR.NZ NotTeclaB     ;em memória, as cores ficam codificadas com números
                    (B:0; P:1; E:2; V:3; Z:4; A:5)
                    MOV R1, 'B'
                    MOV R4, BRANCO
                    MOV M[R3], R4
                    JMP EscreverCor
        NotTeclaB:  CMP R1, 'p'
                    BR.NZ NotTeclaP
                    MOV R1, 'P'
                    MOV R4, PRETO
                    MOV M[R3], R4
                    JMP EscreverCor
        NotTeclaP:  CMP R1, 'e'
                    BR.NZ NotTeclaE
                    MOV R1, 'E'
                    MOV R4, ENCARNADO
                    MOV M[R3], R4
                    JMP EscreverCor
        NotTeclaE:  CMP R1, 'v'
                    BR.NZ NotTeclaV
                    MOV R1, 'V'
                    MOV R4, VERDE
                    MOV M[R3], R4
                    BR EscreverCor
        NotTeclaV:  CMP R1, 'z'
                    BR.NZ NotTeclaZ
                    MOV R1, 'Z'
                    MOV R4, AZUL
                    MOV M[R3], R4
                    BR EscreverCor
        NotTeclaZ:  CMP R1, 'a'
                    JMP.NZ Espera       ;caso a tecla não seja nenhuma das cores
                    autorizadas, o programa volta ao modo de espera
                    MOV R1, 'A'
                    MOV R4, AMARELO
                    MOV M[R3], R4
                    BR EscreverCor
    EscreverCor:    MOV M[IO_CURSOR], R5        ;escreve no ecra cada cor inserida pelo
    jogador
                    CALL EscCar
                    ADD R5, 4 ;numero de espaços entre as cores no ecra
```

```
                    INC R2
                    CMP R2, M[NumeroCoresSequencia] ;numero de cores da sequencia
                    JMP.N Espera
Fim_EsperarTentativa:POP R5
                    POP R4
                    POP R3
                    POP R2
                    POP R1
                    RETN 1

;Rotina que inicia/reinicia a contagem decrescente do temporizador
IniciarContagem: PUSH R1
                 MOV R1, M[TempoJogo]
                 MOV M[ContagemTempo], R1
                 CALL EscreverDisp7Seg
                 MOV R1, 10
                 MOV M[TIMER_COUNT], R1; 10*100ms = 1s
                 MOV R1, 1
                 MOV M[TIMER_START], R1
                 POP R1
                 RET

;Rotina que para a contagem do temporizador
PararContagem:  MOV M[TIMER_START], R0
                MOV M[ContagemTempo], R0
                CALL EscreverDisp7Seg
                RET

;Rotina que provoca um ligeiro atraso
Delay:      PUSH R1
            MOV R1, R0
CicloDelay: INC R1
            CMP R1, ffffh
            BR.NZ CicloDelay
            POP R1
            RET
;Rotina que provoca um atraso maior
DelayMaior: CALL Delay
            CALL Delay
            CALL Delay
            CALL Delay
            CALL Delay
            CALL Delay
            RET

;Rotina que escreve no diplay de 7 segmentos o valor atual da contagem decrescente
EscreverDisp7Seg:   PUSH R1
                    PUSH R2
                    MOV R1, M[ContagemTempo]
                    MOV R2, 60
                    DIV R1, R2
                    MOV M[DISPLAY_2], R1  ; ContagemTempo/60 - valor dos minutos
                    MOV R1, 10
                    DIV R2, R1
                    MOV M[DISPLAY_1], R2 ;ContagemTempo%60 /10 -> algarismo mais
                    sinificativo dos segundos
                    MOV M[DISPLAY_0], R1 ;ContagemTempo%60 %10 -> algarismo menos
                    sinificativo dos segundos
                    POP R2
                    POP R1
                    RET

;Rotina chamada quando um jogo acaba (mostra o vencedor e chama a rotina que mostra as
estísticas no LCD)
        FimJogo:    PUSH R1
                    PUSH R2
                    PUSH VarTextoR  ;mostrar vencedor no ecra
                    PUSH 1400h
                    CALL EscString
                    PUSH VarTextoQ  ;mensagem de novo jogo
                    PUSH 1500h
```

```
                    CALL EscString
                    CMP M[MelhorJogador], R0
                    BR.NZ ExisteVencedor
                    PUSH VarTextoS      ; Quando não existe vencedor
                    PUSH 140ah
                    CALL EscString
                    MOV M[IniciarNovoJogo], R0
                    BR Estatisticas
ExisteVencedor:     MOV R1, 1412h       ;Quando existe vencedor
                    MOV M[IO_CURSOR], R1
                    MOV R1, M[MelhorJogador]
                    ADD R1, '0'
                    CALL EscCar
                    MOV M[IniciarNovoJogo], R0
    Estatisticas:   CALL MostrarEstatisticas ;;mostrar estatisticas no LCD; este ciclo só
    termina se o jogador clicar no botão 1
                    CMP M[IniciarNovoJogo], R0
                    BR.Z Estatisticas
                    POP R2
                    POP R1
                    RET

;Rotina que mostra a mensagem de vitória
MostrarVitoria:     PUSH R1
                    PUSH R2
                    PUSH VarTextoK  ;mensagem de vitória
                    PUSH 1212h
                    CALL EscString
                    MOV R1, M[NumJogadores]
                    CMP M[JogadorAtual], R1
                    BR.NZ MensagemNovoJogador1  ;se o jogador atual não for o ultimo
                    CALL FimJogo        ;se o jogador atual for o ultimo
                    MOV R1, CicloNovoJogo
                    MOV M[SP + 3], R1 ;alteração do endreço de retorno da rotina
                    BR FimMostrarVitoria
MensagemNovoJogador1:PUSH VarTextoP ;mensagem de proximo jogador
                    PUSH 1400h
                    CALL EscString
                    CALL Leds_vitoria   ;faz piscar os leds verdes
                    CALL Leds_Stop
FimMostrarVitoria:  MOV M[IO_READ], R0  ;limpa o valor introduzido no teclado
                    POP R2
                    POP R1
                    RET


;Rotina que faz piscar os LEDs verdes após uma vitória
Leds_vitoria:       PUSH R1
                    PUSH R2
                    MOV R2, 00F0h   ;leds verdes
PiscarVerdes:       MOV M[LEDS], R0
                    CALL Delay
                    MOV M[LEDS], R2
                    CALL Delay
                    MOV R1, M[IO_STAT]  ;só sai do ciclo quando o utilizador clicar numa tecla
                    CMP R1, R0
                    BR.Z PiscarVerdes
                    POP R2
                    POP R1
                    RET



;Rotina que apaga os LEDs após passar para outro jogador
Leds_Stop:          MOV M[LEDS], R0
                    RET


;Rotina que mostra a mensagem de derrota
MostrarDerrota:     PUSH R1
                    PUSH R2
```

```
                    PUSH VarTextoJ  ;mensagem de derrota
                    PUSH 1212h
                    CALL EscString
                    MOV R1, M[NumJogadores]
                    CMP M[JogadorAtual], R1
                    BR.NZ MensagemNovoJogador2 ;se o jogador atual não for o ultimo
                    CALL FimJogo        ;se o jogador atual for o ultimo
                    MOV R1, CicloNovoJogo
                    MOV M[SP + 3], R1 ;alteração do endreço de retorno da rotina
                    BR FimMostrarDerrota
MensagemNovoJogador2:PUSH VarTextoP ;mensagem de proximo jogador
                    PUSH 1400h
                    CALL EscString
                    CALL Leds_Derrota   ;faz piscar os leds vermelhos
                    CALL Leds_Stop
FimMostrarDerrota:  MOV M[IO_READ], R0  ;limpa o valor introduzido no teclado
                    POP R2
                    POP R1
                    RET

;Rotina que faz piscar os LEDs vermelhos após uma derrota
Leds_Derrota:       PUSH R1
                    PUSH R2
                    MOV R2, f00fh ;leds vermelhos
PiscarVermelhos:    MOV M[LEDS], R0
                    CALL Delay
                    MOV M[LEDS], R2
                    CALL Delay
                    MOV R1, M[IO_STAT] ;só sai do ciclo quando o utilizador clicar numa tecla
                    CMP R1, R0
                    BR.Z PiscarVermelhos
                    POP R2
                    POP R1
                    RET


;Rotina para gerar a sequência aleatória de peças (algoritmo do enunciado)
GerarSeq:       PUSH R1
                PUSH R2
                PUSH R3
                MOV R3, R0
CicloGerar:     MOV R1, M[RandNum]
                TEST R1, 0001h
                BR.Z MenorBitZero; testa se o bit menos significativo de R1 é zero
                MOV R2, M[MascaraAlgoritmoAleatorio]
                XOR R1, R2
                ROR R1, 1
                BR FimGerar
MenorBitZero:   ROR R1, 1
FimGerar:       MOV M[RandNum], R1
                MOV R2, 6; número de cores
                DIV R1, R2
                MOV R1, R0
                ADD R1, Sequencia
                ADD R1, R3
                MOV M[R1], R2; M[Sequencia + R3] = R2, que é o resto da divisão inteira de
                R1 por R2
                INC R3
                CMP R3, M[NumeroCoresSequencia]
                BR.N CicloGerar
                POP R3
                POP R2
                POP R1
                RET

;Rotina para desenhar o tabuleiro no ecra
GerarTabuleiro: PUSH    R1
                PUSH    R2
                MOV     R2, 0000h ;1° linha
                PUSH    VarTexto1
                PUSH    R2
```

```
                CALL    EscString
                PUSH    VarTextoM ;2° linha
                ADD     R2, 0100h
                PUSH    R2
                CALL    EscString
                MOV     R1, 011ah
                MOV     M[IO_CURSOR], R1
                MOV     R1, M[JogadorAtual]
                ADD     R1, '0'
                CALL    EscCar
                PUSH    VarTexto2   ;3° linha
                ADD     R2, 0100h
                PUSH    R2
                CALL    EscString
                PUSH    VarTexto3   ;4° linha
                ADD     R2, 0100h
                PUSH    R2
                CALL    EscString
                PUSH    VarTexto2   ;5° linha
                ADD     R2, 0100h
                PUSH    R2
                CALL    EscString
                MOV     R1,R0
CicloTabuleiro: INC     R1          ;9 primeiras linhas relativas às tentativas
                ADD     R2, 0100h
                PUSH    VarTexto4
                PUSH    R2
                CALL    EscString
                INC     R2
                MOV     M[IO_CURSOR], R2
                DEC     R2
                ADD     R1, '0'
                CALL    EscCar
                SUB     R1,'0'
                CMP     R1, 9
                BR.N    CicloTabuleiro
                ADD     R2, 0100h       ;ultima linha de tentativas
                PUSH    VarTexto4
                PUSH    R2
                CALL    EscString
                MOV     M[IO_CURSOR], R2
                MOV     R1, '1'
                CALL    EscCar
                INC     R2
                MOV     M[IO_CURSOR], R2
                DEC     R2
                MOV     R1, '0'
                CALL    EscCar
                ADD     R2, 0100h               ;separador e linha para solução
                PUSH    VarTexto2
                PUSH    R2
                CALL    EscString
                ADD     R2, 0100h
                PUSH    VarTexto5
                PUSH    R2
                CALL    EscString
                MOV     R2, 0000h
                MOV     M[IO_CURSOR], R2
                POP     R2
                POP     R1
                RET

;Rotina para mostrar a solução
MostrarSol:     PUSH R1
                PUSH R2
                PUSH R3
                MOV R2,R0
                MOV R1, M[NumeroCoresSequencia]
                MOV R3, 6       ;Número de cores maximo
                SUB R3, R1
                MOV R1, 2       ;Numero de espaços entre as cores/2
```

```
                MUL R1, R3
                ADD R2, R3
                ADD R2, 1013h           ;R2 fica com a posição de onde será escrita a
                primeira cor da solução
                MOV R3,R0
    CicloSol:   MOV R1, R0
                ADD R1, Sequencia   ;Sequencia corresponde à posição de memória do primeiro
                elemento do vetor com a sequência gerada
                ADD R1, R3
                MOV R1, M[R1]
                CMP R1, BRANCO
                BR.NZ NotB
                MOV R1, 'B'
                BR EscreverSolCor
    NotB:       CMP R1, PRETO
                BR.NZ NotP
                MOV R1, 'P'
                BR EscreverSolCor
    NotP:       CMP R1, ENCARNADO
                BR.NZ NotE
                MOV R1, 'E'
                BR EscreverSolCor
    NotE:       CMP R1, VERDE
                BR.NZ NotZ
                MOV R1, 'V'
                BR EscreverSolCor
    NotZ:       CMP R1, AZUL
                BR.NZ NotV
                MOV R1, 'Z'
                BR EscreverSolCor
    NotV:       MOV R1, 'A'
EscreverSolCor: MOV M[IO_CURSOR], R2
                CALL EscCar
                ADD R2, 4   ;numero de espaços entre as cores no ecra
                INC R3
                CMP R3, M[NumeroCoresSequencia]
                JMP.N CicloSol
                POP R3
                POP R2
                POP R1
                RET

;Rotina para avaliar a tentativa atual do jogador (recebe o numero da tentativa pelo stack)
AvaliarTentativa:   PUSH R1
                    PUSH R2
                    PUSH R3
                    PUSH R4
                    PUSH R5
                    MOV R1,R0
        LimparAv:   MOV R2, Avaliacao           ;ciclo que limpa o vetor Avaliacao e iguala
        o vetor Sequencia_Copia ao Sequencia e o TentAtual_Copia ao TentAtual
                    ADD R2, R1
                    MOV M[R2], R0
                    MOV R2, Sequencia
                    MOV R3, Sequencia_Copia
                    MOV R4, TentAtual
                    MOV R5, TentAtual_Copia
                    ADD R2, R1
                    ADD R3, R1
                    ADD R4, R1
                    ADD R5, R1
                    MOV R2, M[R2]
                    MOV M[R3], R2
                    MOV R4, M[R4]
                    MOV M[R5], R4
                    INC R1
                    CMP R1, M[NumeroCoresSequencia]
                    BR.N LimparAv
                    MOV R1, R0
        CicloAv1:   MOV R3, Sequencia_Copia     ;CicloAv1: Avalia se existem peças com cores
        e posições certas
```

```
                    MOV R2, TentAtual_Copia
                    ADD R3, R1
                    ADD R2, R1
                    MOV R5, M[R2]
                    CMP R5, M[R3]
                    BR.Z CorPosCerto
    FimCicloAv1:    INC R1
                    CMP R1, M[NumeroCoresSequencia]
                    BR.N CicloAv1
                    MOV R1, R0
                    MOV R2, R0
                    BR CicloAv2
    CorPosCerto:    MOV R4, VAZIO       ;caso existam peças com cor e posição certa, é
    guardado na posição correspondente do vetor Avaliacao um 'P'
                    MOV M[R3], R4; o numero 6 é usado como um "vazio", para que a cor desse
                    sitio não seja identificada mais que uma vez
                    MOV M[R2], R4
                    MOV R3, Avaliacao
                    ADD R3, R1
                    MOV R4, 'P'
                    MOV M[R3], R4
                    BR FimCicloAv1
        CicloAv2:   MOV R3, Sequencia_Copia     ;CicloAv2: Avalia se existem peças com cores
        certas
                    MOV R4, TentAtual_Copia
                    ADD R4, R2
                    ADD R3, R1
                    MOV R5, M[R3]
                    CMP R5, 6 ;se a posição atual da Sequencia_Copia estiver "vazia", não é
                    necessário verifica-la
                    BR.Z RetomarCicloAv2
                    CMP R5, M[R4]
                    BR.Z CorCerto
                    INC R2
                    CMP R2, M[NumeroCoresSequencia]
                    BR.N CicloAv2
RetomarCicloAv2:    MOV R2, R0
                    INC R1
                    CMP R1, M[NumeroCoresSequencia]
                    BR.N CicloAv2
                    BR FimCicloAv2
        CorCerto:   MOV R5, VAZIO       ;caso existam peças com cor certa, é guardado na
        posição correspondente do vetor Avaliacao um 'B'
                    MOV M[R3], R5
                    MOV M[R4], R5
                    MOV R3, Avaliacao
                    ADD R3, R1
                    MOV R4, 'B'
                    MOV M[R3], R4
                    BR RetomarCicloAv2
    FimCicloAv2:    MOV R1, M[SP + 7]       ;quando a avaliação termina, são escritos nas
    respetivas linhas os resultados, primeiro os 'P' e depois os 'B'
                    MOV R2, 0100h
                    MUL R1, R2
                    ADD R2, 042dh       ;R2 fica com a posição no ecra do sitio onde
                    começará a ser escrita a avaliação
                    MOV R3, R0
                    MOV R4, R0
        CicloAvEscP:MOV R1, Avaliacao
                    ADD R1, R3
                    INC R3
                    CMP R3, M[NumeroCoresSequencia]
                    BR.P CicloAvEscB
                    MOV R1, M[R1]
                    CMP R1, 'P'
                    BR.NZ CicloAvEscP
                    MOV M[IO_CURSOR], R2
                    CALL EscCar
                    INC R2
                    BR CicloAvEscP
        CicloAvEscB:MOV R1, Avaliacao
```

```
                    ADD R1, R4
                    INC R4
                    CMP R4, M[NumeroCoresSequencia]
                    BR.P FimCicloAv
                    MOV R1, M[R1]
                    CMP R1, 'B'
                    BR.NZ CicloAvEscB
                    MOV M[IO_CURSOR], R2
                    CALL EscCar
                    INC R2
                    BR CicloAvEscB
        FimCicloAv: POP     R5
                    POP     R4
                    POP     R3
                    POP     R2
                    POP     R1
                    RETN    1

;Rotina para verificar se o jogador ganhou o jogo (Devolve 1 se ganhou ou 0 se não através
do registo R2)
VerificarVit:   PUSH R1
                PUSH R3
                MOV R2, R0
                MOV R1, R0
    CicloVerif: MOV R3, Avaliacao   ;este ciclo verifica se todos os elementos do vetor
    Avaliacao sao 'P'
                ADD R3, R1
                MOV R3, M[R3]
                CMP R3, 'P'
                BR.NZ FimVerif
                INC R1
                CMP R1, M[NumeroCoresSequencia]
                BR.N CicloVerif
                MOV R2, 1
    FimVerif:   POP R3
                POP R1
                RET

;Verifica se o melhor jogador foi alterado e atualiza-o. Recebe o numero de tentativas da
jogada atual pelo R1
AtualizarMelhorJogador: CMP R1, M[MelhorJogadasAcerto]
                        BR.N NovoMelhorJogador
                        RET
NovoMelhorJogador:      PUSH R2
                        MOV R2, M[JogadorAtual]
                        MOV M[MelhorJogador], R2
                        MOV M[MelhorJogadasAcerto], R1
                        POP R2
                        RET

;Limpa as estatisticas do jogo anterior
ResetEstatiticas:   PUSH R1
                    MOV R1, 0fffh
                    MOV M[MelhorTempoJogada], R1
                    MOV R1, 000fh
                    MOV M[MelhorJogadasAcerto], R1
                    MOV M[JogadorAtual], R0
                    MOV M[AccTemposJogo], R0
                    MOV M[MelhorJogador], R0
                    MOV M[AccJogadasAcerto], R0
                    MOV M[NumVitorias], R0
                    POP R1
                    RET

;Rotina que atualiza as estatisticas de jogo ;Recebe o valor atual de tentativas pelo R1
AtualizarEstatisticas_Jogo: PUSH R2
                            ADD M[AccJogadasAcerto], R1
                            INC M[NumVitorias]
                            MOV R2, M[TempoTotalJogada]
                            ADD M[AccTemposJogo], R2
                            CMP R2, M[MelhorTempoJogada]
```

```
                        BR.N NovoMelhorTempo
                        POP R2
                        RET
        NovoMelhorTempo:    MOV M[MelhorTempoJogada],R2
                        POP R2
                        RET

;Rotina que atualiza as estatisticas do jogador ;Recebe o valor atual de tentativas pelo R1
AtualizarEstatisticas_Jogador:  PUSH R3
                                PUSH R2
                                MOV R3, M[JogadorAtual]
                                DEC R3  ;R3 fica com o indice da posição correspondente ao
                                jogador atual nos vetores
                                ADD M[R3 + AccJogadasAcerto_Jogador], R1
                                INC M[R3 + NumVitorias_Jogador]
                                MOV R2, M[TempoTotalJogada]
                                ADD M[R3 + AccTemposJogo_Jogador], R2
                                CMP R2, M[R3 + MelhorTempoJogo_Jogador]
                                BR.N NovoMelhorTempo_Jogador
                                POP R2
                                POP R3
                                RET
NovoMelhorTempo_Jogador:        MOV M[R3 + MelhorTempoJogo_Jogador],R2
                                POP R2
                                POP R3
                                RET




;Rotina que escreve um tempo na segunda linha do LCD no formato minutos e segundos. Este
tempo é passado pelo STACK em segundos
EscreverTempoLCD:   PUSH R1
                    PUSH R2
                    PUSH R3
                    MOV R1, M[SP + 5]
                    MOV R2, 60
                    DIV R1, R2
                    MOV R2, 10
                    DIV R1, R2
                    MOV R3, 8010h
                    MOV M[LCD_CONTROL], R3
                    ADD R1, '0'
                    MOV M[LCD_WRITE], R1  ; Tempo/60 /10 - algarismo mais significativo dos
                    minutos
                    INC R3
                    MOV M[LCD_CONTROL],R3
                    ADD R2, '0'
                    MOV M[LCD_WRITE], R2  ; Tempo/60 %10 - algarismo menos significativo
                    dos minutos
                    INC R3
                    MOV M[LCD_CONTROL],R3
                    MOV R1, 'm'
                    MOV M[LCD_WRITE], R1
                    MOV R1, M[SP + 5]
                    MOV R2, 60
                    DIV R1, R2
                    MOV R1, 10
                    DIV R2, R1
                    INC R3
                    MOV M[LCD_CONTROL],R3
                    ADD R2, '0'
                    MOV M[LCD_WRITE], R2  ; Tempo/60 /10 - algarismo mais significativo dos
                    segundos
                    INC R3
                    MOV M[LCD_CONTROL],R3
                    ADD R1, '0'
                    MOV M[LCD_WRITE], R1  ; Tempo/60 %10 - algarismo menos significativo
                    dos segundos
```

```
                    INC R3
                    MOV M[LCD_CONTROL],R3
                    MOV R1, 's'
                    MOV M[LCD_WRITE], R1
                    POP R3
                    POP R2
                    POP R1
                    RETN 1

;Rotina que escreve um tempo na janela no formato minutos e segundos. Este tempo é passado
pelo STACK em segundos, bem como
;a posição onde será escrito
EscreverTempoJanela: PUSH R1
                    PUSH R2
                    PUSH R3
                    MOV R1, M[SP + 6] ;Tempo em segundos
                    MOV R2, 60
                    DIV R1, R2
                    MOV R2, 10
                    DIV R1, R2
                    MOV R3, M[SP + 5];posição no ecrã
                    MOV M[IO_CURSOR], R3
                    ADD R1, '0'
                    MOV M[IO_WRITE], R1  ; Tempo/60 /10 - algarismo mais significativo dos
                    minutos
                    INC R3
                    MOV M[IO_CURSOR],R3
                    ADD R2, '0'
                    MOV M[IO_WRITE], R2  ; Tempo/60 %10 - algarismo menos significativo dos
                    minutos
                    INC R3
                    MOV M[IO_CURSOR],R3
                    MOV R1, 'm'
                    MOV M[IO_WRITE], R1
                    MOV R1, M[SP + 6]
                    MOV R2, 60
                    DIV R1, R2
                    MOV R1, 10
                    DIV R2, R1
                    INC R3
                    MOV M[IO_CURSOR],R3
                    ADD R2, '0'
                    MOV M[IO_WRITE], R2  ; Tempo/60 /10 - algarismo mais significativo dos
                    segundos
                    INC R3
                    MOV M[IO_CURSOR],R3
                    ADD R1, '0'
                    MOV M[IO_WRITE], R1  ; Tempo/60 %10 - algarismo menos significativo dos
                    segundos
                    INC R3
                    MOV M[IO_CURSOR],R3
                    MOV R1, 's'
                    MOV M[IO_WRITE], R1
                    POP R3
                    POP R2
                    POP R1
                    RETN 2

;Rotina que mostra as estatisticas do jogo no LCD
MostrarEstatisticas: PUSH R1
                    PUSH R2
                    PUSH R3
                    CALL LimparLCD
                    CMP M[NumVitorias], R0
                    JMP.Z  ZeroVitorias    ;caso nenhum dos jogadores tenha terminado o jogo
                    PUSH VarTextoT     ;Melhor tempo de jogo
                    PUSH 0000h
                    CALL EscStringLCD
                    PUSH M[MelhorTempoJogada]
                    CALL EscreverTempoLCD
                    CALL DelayMaior
```

```
                        CALL LimparLCD
                        PUSH VarTextoU          ;Média tempos de jogo
                        PUSH 0000h
                        CALL EscStringLCD
                        MOV R1, M[AccTemposJogo]
                        MOV R2, M[NumVitorias]
                        DIV R1, R2
                        PUSH R1
                        CALL EscreverTempoLCD
                        CALL DelayMaior
                        CALL LimparLCD
                        PUSH VarTextoV          ;Média jogadas de acerto
                        PUSH 0000h
                        CALL EscStringLCD
                        MOV R1, M[AccJogadasAcerto]
                        MOV R2, M[NumVitorias]
                        DIV R1, R2
                        CMP R1, 10
                        JMP.Z MediaDoisDigitos ;caso a média seja 10
                        ADD R1, '0'
                        MOV R2, 801bh
                        MOV M[LCD_CONTROL], R2
                        CALL EscCarLCD          ;parte interira da média = N°total de
                        jogadas/n°vitórias
                        INC R2
                        MOV M[LCD_CONTROL], R2
                        MOV R1, '.'
                        CALL EscCarLCD
                        INC R2
                        MOV M[LCD_CONTROL], R2
                        MOV R1, M[AccJogadasAcerto]
                        MOV R2, M[NumVitorias]
                        DIV R1, R2
                        MOV R1,10
                        MUL R1, R2
                        MOV R1, M[NumVitorias]
                        DIV R2, R1
                        MOV R1, R2
                        ADD R1, '0'
                        CALL EscCarLCD ;primeiro digito decimal da média = (N°total de
                        jogadas%n°vitórias * 10)/n°vitorias
                        CALL DelayMaior
                        JMP Fim_MostrarEstatisticas
MediaDoisDigitos:               MOV R2, 801bh
                                MOV M[LCD_CONTROL], R2
                                MOV R1, '1'
                                CALL EscCarLCD
                                INC R2
                                MOV M[LCD_CONTROL], R2
                                MOV R1, '0'
                                CALL EscCarLCD
                                INC R2
                                MOV M[LCD_CONTROL], R2
                                MOV R1, '.'
                                CALL EscCarLCD
                                INC R2
                                MOV M[LCD_CONTROL], R2
                                MOV R1, '0'
                                CALL EscCarLCD
                                CALL DelayMaior
                                BR Fim_MostrarEstatisticas
ZeroVitorias:                   PUSH VarTextoZ      ;Jogo sem vitorias
                                PUSH 0000h
                                CALL EscStringLCD
                                CALL DelayMaior
Fim_MostrarEstatisticas:        POP R3
                                POP R2
                                POP R1
                                RET

;Rotina que mostra as estatisticas do jogador no ecra
```

```
MostrarEstatisticas_Jogador: PUSH R1
                             PUSH R2
                             PUSH R3
                             PUSH R4
                             MOV R4, M[JogadorAtual]
                             DEC R4 ;R4 fica com o indice da posição correspondente ao
                             jogador atual nos vetores
                             PUSH VarTextoX     ;Estatisticas do jogador
                             PUSH 0030h
                             CALL EscString
                             CMP M[R4 + NumVitorias_Jogador], R0
                             JMP.Z  ZeroVitorias_Jogador    ;caso o jogador nao tenha vitórias
                             PUSH VarTextoT     ;Melhor tempo de jogo
                             PUSH 0130h
                             CALL EscString
                             PUSH M[R4 + MelhorTempoJogo_Jogador]
                             PUSH 013eh
                             CALL EscreverTempoJanela
                             PUSH VarTextoU         ;Média tempos de jogo
                             PUSH 0230h
                             CALL EscString
                             MOV R1, M[R4 + AccTemposJogo_Jogador]
                             MOV R2, M[R4 + NumVitorias_Jogador]
                             DIV R1, R2
                             PUSH R1
                             PUSH 023dh
                             CALL EscreverTempoJanela
                             PUSH VarTextoV         ;Média jogadas de acerto
                             PUSH 0330h
                             CALL EscString
                             MOV R1, M[R4 + AccJogadasAcerto_Jogador]
                             MOV R2, M[R4 + NumVitorias_Jogador]
                             DIV R1, R2
                             CMP R1, 10     ;Caso o a média seja 10(2 digitos)
                             JMP.Z MediaDoisDigitos_Jogador
                             ADD R1, '0'
                             MOV R2, 034bh
                             MOV M[IO_CURSOR], R2
                             CALL EscCar            ;parte interira da média = N°total de
                             jogadas/n°vitórias
                             INC R2
                             MOV M[IO_CURSOR], R2
                             MOV R1, '.'
                             CALL EscCar
                             INC R2
                             MOV M[IO_CURSOR], R2
                             MOV R1, M[R4 + AccJogadasAcerto_Jogador]
                             MOV R2, M[R4 + NumVitorias_Jogador]
                             DIV R1, R2
                             MOV R1,10
                             MUL R1, R2
                             MOV R1, M[R4 + NumVitorias_Jogador]
                             DIV R2, R1
                             MOV R1, R2
                             ADD R1, '0'
                             CALL EscCar ;primeiro digito decimal da média = (N°total de
                             jogadas%n°vitórias * 10)/n°vitorias
                             JMP Fim_MostrarEstatisticas_Jogador
MediaDoisDigitos_Jogador:    MOV R2, 034bh
                             MOV M[IO_CURSOR], R2
                             MOV R1, '1'
                             CALL EscCar
                             INC R2
                             MOV M[IO_CURSOR], R2
                             MOV R1, '0'
                             CALL EscCar
                             INC R2
                             MOV M[IO_CURSOR], R2
                             MOV R1, '.'
                             CALL EscCar
                             INC R2
```

```
                            MOV M[IO_CURSOR], R2
                            MOV R1, '0'
                            CALL EscCar
                            BR Fim_MostrarEstatisticas_Jogador
ZeroVitorias_Jogador:       PUSH VarTextoY     ;Jogador sem vitorias
                            PUSH 0130h
                            CALL EscString
Fim_MostrarEstatisticas_Jogador:    POP R4
                                    POP R3
                                    POP R2
                                    POP R1
                                    RET


;Rotina para escrever uma string na janela de texto (recebe o endreço da string e a posiçao
através do stack)
EscString:      PUSH    R1
                PUSH    R2
                PUSH    R3
                MOV     R2, M[SP+6]    ; Apontador para inicio da "string"
                MOV     R3, M[SP+5]    ; Localizacao do primeiro carater
    Ciclo:      MOV     M[IO_CURSOR], R3
                MOV     R1, M[R2]
                CMP     R1, FIM_TEXTO
                BR.Z    FimEsc
                CALL    EscCar
                INC     R2
                INC     R3
                BR      Ciclo
    FimEsc:     POP     R3
                POP     R2
                POP     R1
                RETN    2               ; Actualiza STACK

;Rotina para escrever uma string no LCD (recebe o endreço da string e a posiçao através do
stack)
EscStringLCD:   PUSH    R1
                PUSH    R2
                PUSH    R3
                MOV     R2, M[SP+6]    ; Apontador para inicio da "string"
                MOV     R3, M[SP+5]    ; Localizacao do primeiro carater
                OR      R3, 8000h
    CicloLCD:   MOV     M[LCD_CONTROL], R3
                MOV     R1, M[R2]
                CMP     R1, FIM_TEXTO
                BR.Z    FimEscLCD
                CALL    EscCarLCD
                INC     R2
                INC     R3
                BR      CicloLCD
    FimEscLCD:  POP     R3
                POP     R2
                POP     R1
                RETN    2               ; Actualiza STACK


;Rotina para escrever um caracter na janela de texto (recebe o carater através de
R1)
EscCar:         MOV     M[IO_WRITE], R1
                RET

;Rotina para escrever um caracter na janela de texto (recebe o carater através de
R1)
EscCarLCD:      MOV     M[LCD_WRITE], R1
                RET

;Programa
Inicio:         MOV R1, SP_INIC ;configurar o stack
                MOV SP, R1
                MOV R1, RotinaInt0
                MOV M[TAB_INT0], R1     ;configurar a interrupção 0 (botão 0)
```

```
                MOV R1, RotinaInt1
                MOV M[TAB_INT1], R1
                MOV R1, RotinaInt15     ;configurar a interrupção 1 (botão 1)
                MOV M[TAB_INT15], R1
                MOV R1,8003h            ;configuração da máscara de interrupções (permite a
                int15 - temporizador e int0 - btn0 e int1- btn1)
                MOV M[MASCARA_INT], R1  ;configurar mascara de interrupções
                CALL Leds_Stop
                CALL LimparDisp7Seg
                CALL LimparJanela
                CALL LimparLCD
                CALL EcraInicial        ;Mostra o ecra inicial e gera a semente para o
                algoritmo aleatório
                CALL LimparJanela
                CALL SegundoEcra
                CALL LimparJanela
                ENI                     ;Ativa as interrupções
CicloNovoJogo:  CALL ResetEstatiticas
CicloJogadores: MOV R1, R0
                INC M[JogadorAtual]
                MOV M[TempoTotalJogada], R0
                MOV M[Desistir], R0
                CALL LimparLCD
                CALL LimparJanela       ;Limpa a janela de texto
                CALL GerarSeq           ;gera a sequência aleatória de cores
                CALL GerarTabuleiro     ;desenha o tabuleiro na janela de texto
                ;CALL MostrarSol
CicloTentativas:INC R1                  ;ciclo que recebe as tentativas do jogador e as avalia
                CMP R1, 10
                BR.P Derrota
                CALL IniciarContagem    ;inicia/reinicia a contagem do temporizador
                PUSH R1                 ;parametro para a rotina EsperarTentativa
                CALL EsperarTentativa
                CMP M[Desistir], R0
                BR.NZ Derrota
                PUSH R1                 ;parametro para a rotina AvaliarTentativa
                CALL AvaliarTentativa
                CALL VerificarVit       ;Verifica se o jogador ganhou o jogo e passa pelo R2
                essa informação (1-ganhou, 0-nao ganhou)
                CMP R2, 1
                BR.Z Vitoria
                BR  CicloTentativas
Derrota:        CALL ParарContagem      ;pára a contagem do temporizador
                CALL MostrarSol
                CALL MostrarEstatisticas_Jogador
                CALL MostrarDerrota
                MOV R3, M[NumJogadores]
                CMP R3, M[JogadorAtual]
                JMP.NZ CicloJogadores       ;Se o jogador atual for o último, é iniciado um
                novo jogo
                BR FIM
Vitoria:        CALL ParарContagem
                CALL MostrarSol
                CALL AtualizarMelhorJogador     ;verifica se o jogador atual completou em
                menos jogadas do que algum dos anteriores
                CALL AtualizarEstatisticas_Jogo             ;as estatisticas só incidem
                sobre jogadores que acertaram na chave
                CALL AtualizarEstatisticas_Jogador
                CALL MostrarEstatisticas_Jogador
                CALL MostrarVitoria
                MOV R3, M[NumJogadores]
                CMP R3, M[JogadorAtual]
                JMP.NZ CicloJogadores       ;Se o jogador atual for o último, é iniciado um
                novo jogo
FIM:            BR FIM
```