Esta prova tem 6 questões, num total de 200 pontos. Fundamente todas as respostas.

Sistemas de entrada/saída

[20] 1. Pretende-se ler um ficheiro de 320 kB armazenado num disco de 6000 RPM com tempo médio de busca de 4,9 ms e taxa de transferência de 40 MB/s.

Considerando que cada cilindro do disco possui 128 setores de 4 kB indique, justificando, entre que valores pode variar o tempo que demora a ler o referido ficheiro.

(Nota: para simplificar os cálculos considere 1 kB = 1×10^3 B e 1 MB = 1×10^6 B.)

Resposta: Alguns cálculos preparatórios:

- 320 kB = 80 setores;
- Tempo de meia volta: $0{,}5 \times 60/6000 = 5$ ms.
- Tempo de transferência de um setor: 4 kB/40 MB/s = 0,1 ms.

Tempo de leitura do ficheiro

Pior caso (setores aleatórios): $T = 80 \times (4{,}9\,\text{ms} + 5\,\text{ms} + 0{,}1\,\text{ms}) = 800\,\text{ms}$

Melhor caso (setores consecutivos num único cilindro): $T = 4{,}9\,\text{ms} + 5\,\text{ms} + 80 \times 0{,}1\,\text{ms} = 17{,}9\,\text{ms}$

2. Uma aplicação de gestão é executada num computador com as seguintes características: CPU de 3600 MIPS, barramento de 800 MB/s, controlador de discos com taxa de transferência máxima de 240 MB/s e capacidade para 4 discos com acesso simultâneo. O computador tem instalado um único disco SSD com uma taxa de transferência de 80 MB/s e setores de 16 kB. Cada operação de E/S necessita de 360000 instruções do CPU.

[15] (a) Indique, justificando, qual o componente que está a limitar o desempenho do sistema dado pelo número de operações de E/S por segundo.

Resposta: O CPU é capaz de executar 3600 M / 360 k = 10 k E/S por segundo.

O barramento suporta até 800 M / 16 k = 50 k E/S por segundo.

O controlador de discos suporta até 240 M / 16 k = 15 k E/S por segundo.

Um acesso ao disco SSD demora 16 k / 80 M = 0,2 ms, ou seja, o disco permite 5 k E/S por segundo.

Conclusão: o elemento que está a limitar o desempenho é o disco (seguido do CPU).

[5] (b) Apresente e justifique qual a medida mais indicada para melhorar o desempenho do sistema.

> **Resposta:** O primeiro passo para melhorar o desempenho será acrescentar um segundo disco por forma a subir a capacidade para 10 k E/S por segundo, que é o máximo que o CPU atual aguenta.

Análise de código

3. Considere o seguinte programa constituído por dois ficheiros.

```
// Ficheiro prog.c

extern "C" unsigned int rotina(int valores[32]);
main()
{
  int v[32]={-1, 2, 3, 1, -2, -3, 1, 0,
             0, 0, 0, 0,  0,  0, 0, 0,
             0, 0, 0, 0,  0,  0, 0, 0,
             0, 0, 0, 0,  0,  0, 0, 0};
  unsigned int res;
  res = rotina(v);
  std::cout << "Resultado: " << res << std::endl;
  return 0;
}
```

```
1  ;; Ficheiro rotinas.asm
2  aux PROC P:PTR SDWORD
3          xor     eax, eax
4          mov     edx, P
5          mov     ecx, [edx]
6          cmp     ecx, [edx - type SDWORD]
7          jl      E1
8          jg      E2
9          jmp     fim
10 E1:     cmp     ecx, [edx + type SDWORD]
11         jle     fim
12         jmp     E3
13 E2:     cmp     ecx, [edx + type SDWORD]
14         jge     fim
15 E3:     inc     eax
16 fim:    ret
17 aux ENDP
18
19 rotina  PROC C  USES esi ebx valores: PTR SDWORD
20         xor     ebx, ebx
21         mov     ecx, 30
22         mov     esi, valores
```

```
23          add     esi, type SDWORD
24  L1:     push    ecx
25          invoke  aux, esi
26          .IF     eax == 1
27            stc
28          .ELSE
29            clc
30          .ENDIF
31          rcr     ebx, 1
32          add     esi, type SDWORD
33          pop     ecx
34          loop    L1
35          ror     ebx, 1
36          mov     eax, ebx
37          ret
38  rotina  ENDP
```

A sub-rotina `aux` determina se o número guardado na posição `P` está estritamente contido entre os valores dos seus dois vizinhos em memória.

[10] (a) Considere *as três primeiras invocações* da sub-rotina `aux`, realizadas a partir da linha 25. Para cada uma das invocações, indique o valor de `ecx` em `aux` e classifique todos os saltos condicionais existentes na sub-rotina. Cada salto condicional deve ser classificado como **tomado** (se a condição for verdadeira), **não tomado** (se a condição for falsa) ou **não executado**.

Resposta:

Três primeiras invocações de `aux`:

`ecx`	linha 7 (`jl`)	linha 8 (`jg`)	linha 11 (`jle`)	linha 14 (`jge`)
2	não tomado	tomado	não executado	não tomado
3	não tomado	tomado	não executado	tomado
1	tomado	não executado	não tomado	não executado

[10] (b) Explique a razão para a utilização do par de instruções `push ecx / pop ecx` na sub-rotina `rotina` (linhas 24 e 33).

Resposta: De acordo com a convenção de invocação de sub-rotinas `stdcall`, o valor do registo `ecx` pode ser alterado em qualquer sub-rotina. Como o valor de `ecx` é necessário para controlar o ciclo existente em `rotina` (instrução `loop`), é preciso garantir que qualquer alteração do valor feita por `aux` não afeta a execução de `rotina`, o que é feito repondo o valor anterior à invocação. Para isso, o valor de `ecx` é guardado na pilha antes da invocação de `aux` e retirado depois da invocação e antes da execução da instrução `loop`.

[10] (c) Indique o que é apresentado no monitor e explique a tarefa realizada pelo programa.

Resposta:

No monitor aparece: **Resultado: 26**

Conforme indicadao no enunciado, a sub-rotina `aux` determina se o valor apontado por `P` está estritamente contido entre os seus dois vizinhos. No caso afirmativo a sub-rotina retorna o valor 1; no caso contrário, retorna o valor 0.

A sub-rotina `rotina` invoca `aux` para todos os elementos da sequência `valores`, exceto o primeiro e o último. Sempre que `aux` retorna 1, é incluído um 1 na sequência de bits guardada em `ebx`; no caso contrário, é incluído um 0. A inclusão do bit é feita por uma rotação que envolve o indicador de transporte *(carry flag)*.

O registo `ebx` tem 32 bits, que é igual ao número de elementos de `valores`. No fim, o bit de `ebx` correspondente a cada elemento da sequência está a 1 ou a 0 conforme o resultado de `aux`.

Para a sequência do enunciado, o padrão binário é `0000 0000 0000 0000 0000 0000 0001 1010`, que corresponde a 26_{10}, o valor apresentado no monitor.

Os três valores a 1 correspondem a:

$-1 < \mathbf{2} < 3$ $\quad$ $3 > \mathbf{1} > -2$ $\quad$ $1 > \mathbf{-2} > -3$

(d) Considere a primeira chamada da sub-rotina `aux` e assuma que antes da execução da instrução da linha 5 o valor de `ESP` é `07000400H` e que o estado da pilha é o indicado abaixo (valores em hexadecimal):

Endereço	Conteúdo
07000414	01001000
07000410	10001000
0700040C	?
07000408	00600500
07000404	0040F000
07000400	00abcd00

[5] i. Qual é o valor do elemento da pilha situado no endereço de memória `0700040CH`? Justifique.

[5] ii. Qual é o endereço em memória de `v[3]` (elemento da sequência declarada na função `main`)? Justifique.

Resposta: i. Para a primeira invocação de Os valores residentes na pilha são (a partir do topo, i.e., de baixo para cima): `EBP` anterior, endereço de retorno, apontador para `valores[1]`, valor de `ECX` colocado na pilha antes da invocação de `aux`, `EBX` anterior (do prólogo de `rotina`), `ESI` anterior (do prólogo de `rotina`).

Logo, o valor pedido é o valor de `ECX`, que é `0000001EH = 30`.

ii. Neste caso, a sequência `valores` (em `rotina`) corresponde à sequência `v` (em `main`). Conforme indicado na solução da alínea anterior, o endereço de `valores[1]` é `00600500H`. O elemento `v[3]` fica 8 bytes mais à frente (dois elementos, 4 bytes por elemento), logo o seu endereço é `00600508H`.

Programação

4. Um camião frigorífico para distribuição alimentar possui um sistema que regista em memória, de minuto a minuto, a temperatura no interior da câmara frigorífica que deverá ser sempre inferior a $-20\,^{\circ}\mathrm{C}$. O sistema é ligado de manhã quando o camião está em condições de sair do armazém e é desligado ao fim do dia quando o camião regressa ao armazém.

[5] (a) Assumindo que se quer registar valores inteiros da temperatura durante 12h, indique, justificando, o tipo de variável mais adequado por forma a minimizar o espaço ocupado em memória. Declare num segmento de dados a sequência `temperaturas`, desse tipo, por forma a inicializar os seus elementos a $-100\,^{\circ}\mathrm{C}$.

Resposta: Nesta aplicação os valores de temperatura a armazenar nunca ultrapassarão a gama [-128, +127[por isso o tipo `sbyte` (8 bit com sinal) será a escolha mais eficiente em termos de minimização do espaço de memória ocupado. A declaração pedida será então:

```
.data
temperaturas sbyte 720 dup (-100)        ; 12 horas = 720 minutos
```

[20] (b) No fim do dia pretende-se saber quantas vezes a temperatura subiu acima de um dado limiar de temperatura `tx`. Apresente a sub-rotina `alarmes` que retorna esse valor. Por exemplo, se `tx=0` e `temperaturas` tiver o conteúdo ⟨-2,-1,1,3,1,-1,-2,1,1,2⟩, a sub-rotina deve retornar o valor 2.

O protótipo da sub-rotina é:

```
alarmes proto tx:sbyte
```

Resposta:

```
.code
alarmes proc uses esi ebx tx:sbyte
     mov esi, offset temperaturas
     mov ecx, lengthof temperaturas
     mov ah, tx
     xor ebx, ebx  ; ebx=0 significa temp. abaixo do limiar
     xor edx, edx  ; número de passagens acima do limiar

@@:  lodsb
     cmp al, -100  ; para o caso de terminar
     jz  fim       ;  antes de atingir 12h de registo
     .if ( sbyte ptr al > ah ) && (ebx == 0)
         inc edx
         inc ebx          ; passou acima do limiar
     .elseif ( sbyte ptr al <= ah )
         xor ebx, ebx     ; voltou a descer abaixo do limiar
     .endif
     loop @b
fim: mov eax,edx  ; total de passagens acima do limiar
     ret
alarmes endp
```

[15] (c) Pretende-se também saber que percentagem (inteira) de tempo a temperatura esteve abaixo de um dado limiar de temperatura `tx`. Apresente a sub-rotina `abaixo` que retorna esse valor. Por exemplo, se a temperatura esteve abaixo do limiar durante 2/3 do tempo, a sub-rotina deve retornar **66** (a parte decimal é ignorada). O protótipo da sub-rotina é:

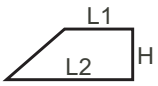
```
abaixo proto tx:sbyte
```

Resposta:

```
.code
abaixo proto uses esi ebx tx:sbyte
     mov esi, offset temperaturas
     mov ecx, lengthof temperaturas
     mov ah, tx
     xor ebx, ebx                ; minutos de viagem
     xor edx, edx                ; minutos abaixo do limiar
@@:  lodsb
     cmp al, -100
     jz  fim
     inc ebx
     .if ( sbyte ptr al < ah )
         inc edx
     .endif
     loop @b
fim: mov eax, 100
     mul edx
     xor edx, edx
     div ebx
     ret
abaixo endp
```

5. Pretende-se desenvolver sub-rotinas destinadas a calcular o valor da área de polígonos.

[20] (a) Implemente a sub-rotina `areaTrap` que recebe como parâmetros as dimensões de um trapézio (figura ao lado) e calcula a respetiva área dada por $\frac{L1+L2}{2} \times H$.



O protótipo da sub-rotina é:

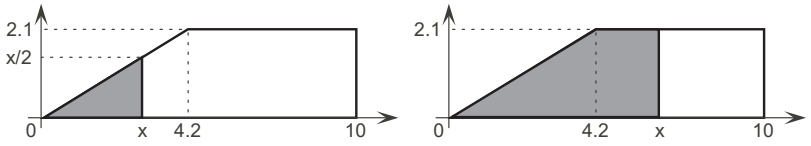
```
areaTrap proto L1:real8, L2:real8, H:real8
```

Resposta:

```
.data
x2_0   real8  2.0
.code
areaTrap proc L1:real8, L2:real8, H:real8
    fld   L1
    fadd  L2
    fdiv  x2_0
    fmul  H
    ret
areaTrap endp
```

[20] (b) Implemente a sub-rotina `areaFig` que calcula a área da região do trapézio, delimitada por `x` ($0 \leq x \leq 10$), mostrada a sombreado na figura. Para o efeito utilize a sub-rotina anterior `areaTrap`.

Note que há a considerar duas situações, ilustradas na figura (`x` ≤ 4.2 à esquerda e `x` > 4.2 à direita).



O protótipo da sub-rotina é: `areaFig proto x:real8`

Resposta:

```
.data
x2_0    real8  2.0
x2_1    real8  2.1
x4_2    real8  4.2

.code
areaFig proc x:real8
local L1:real8, H:real8
    fld  x2_1
    fstp H
    fld  x
    fld  x4_2
    fcomip ST(0), ST(1) ; só x fica na pilha
    jae  tri            ; triângulo (L1=0)
    fsub x4_2           ; ST(0)=x-4.2
    jmp  cal
tri:fld  x              ; ST(0)=x e ST(1)=x
    fdiv x2_0
    fstp H              ; H=x/2
    fsub x              ; ST(0)=0
cal:fstp L1             ; pilha vazia
    invoke areaTrap, L1, x, H
    ret
areaFig endp
```

Escolha múltipla

6. As seguintes questões têm apenas uma resposta correta.

[8] (a) Na sub-rotina `ROT1 PROC v1:WORD, v2:DWORD`, o argumento `v1` pode ser acedido usando:

A. `[EBP+8]` B. `[EBP+6]` C. `[EBP-8]` D. `[EBP-6]`

Resposta: A.

[8] (b) Qual das seguintes sub-rotinas tem o prólogo indicado a seguir?

```
push ebp
mov  ebp,esp
push esi
push ebx
```

A. `rot2 proc p1:dword, p2:dword`
B. `rot3 proc uses ebx esi p1:dword, p2:dword`
C. `rot4 proc p1:word, p2:word`
D. `rot1 proc uses esi ebx p1:word, p2:word`

Resposta: D.

[8] (c) A instrução `REPNZ SCASD` termina, se na sequência a examinar existir pelo menos um elemento:

A. diferente do valor de `EAX`
B. diferente do valor de `EDI`
C. igual ao valor de `EAX`
D. igual ao valor de `EDI`

Resposta: C.

[8] (d) Considere o seguinte fragmento de código:

```
mov  AH, 0abh
mov  AL, 0fh
sub  AH, AL
xor  AH, 0ffh
```

Qual é o valor final do registo `AH` após a execução do código?

A. 63h B. 9ch C. 0a0h D. 0f0h

Resposta: A.

[8] (e) Considere o seguinte fragmento de código:

```
        xor     EAX, EAX
        fld1
        fadd    ST(0), ST(0)
        fld     X
        fcomip  ST(0), ST(1)
        fstp    ST(0)
        jae     L1
        mov     EAX, 5
L1:     ...
```

Qual é o valor de `X` que faz com que o registo `EAX` tenha o valor final 0?

A. `X=-2` B. `X=0` C. `X=1` **D.** `X=2`

Resposta: D.

Fim do enunciado.