

SISTEMAS COMPUTACIONAIS

LICENCIATURA EM ENGENHARIA INFORMÁTICA - 2º ANO

Cotações

- 1) _ 1.25
- 2) _ 1.25
- 3) _ 1.25
- 4) _ 1.25
- 5) _ 1.25
- 6) _ 1.25
- 7) _ 1.25
- 8) _ 1.25
- 9a) _ 1.00
- 9b) _ 1.00
- 10) _ 1.00
- 11a) _ 1.00
- 11b) _ 1.00
- 12a) _ 1.00
- 12b) _ 1.00
- 13a) _ 1.00
- 13b) _ 1.00
- 13c) _ 1.00

PROVA DE AVALIAÇÃO ESCRITA SEM CONSULTA
SEM CONSULTA

QUINTA, 3 DE MARÇO DE 2022

DURAÇÃO: 100 MINUTOS COM TOLERÂNCIA INCLuíDA

NOME: _____

NÚMERO: _____

Atenção: Justifique todas as suas respostas na medida do necessário, sem escrever testamentos. Pense, organize as ideias e escreva. Pode apresentar exemplos e/ou grafismos para melhor elucidar as suas respostas. Considere $1\text{TiB}=2^{40}$ Bytes, $1\text{GiB}=2^{30}$ Bytes, $1\text{MiB}=2^{20}$ Bytes e $1\text{kiB}=2^{10}$ Bytes. Nas questões de código, pode dividir a zona de resposta a meio para ter o dobro de linhas.

TEÓRICA / PARTE 1

- 1) Considerando a arquitetura ARMv7, indique 2 registos com funções / tarefas específicas (não genéricas).

- 2) Apresente dois aspetos negativos da passagem de parâmetros por registos.

- 3) Apresente os passos na invocação e regresso de uma sub-rotina, nomeadamente indicando que registos e instruções estão envolvidos? Não incluir os aspetos específicos do método de passagem de parâmetros.

- 4) Apresente a sequência de passos que devem ser tidos em conta numa rotina de atendimento a uma interrupção.

TEÓRICA / PARTE 2

- 5) Indique o que significa “*cache miss*” memória *cache*? Qual o funcionamento da *cache* nessa situação?

- 6) Indique qual a funcionalidade do bit de paridade na comunicação série.

7) Afirmação: “É possível criar programas em *assembly* e utilizar funções da biblioteca standard de C”. Descreva um exemplo disso, com passagem de parâmetros.

8) O que é o System Dispatcher do Sistema Operativo, utilizando *assembly*?

PRÁTICA / ASSEMBLY

9) Responda às questões abaixo, tendo em conta o código *assembly* ARMv7 apresentado:

Código:

```

;R0 with a value
;R1, R2, R3 with STRING_1, STRING_2 and STRING_3 address
STRING1  cmp     r0, #1
          bne     STRING2
          mov     r0, r1
          bal     ENDCHECK
STRING2  cmp     r0, #2
          bne     STRING3
          mov     r0, r2
          bal     ENDCHECK
STRING3  cmp     r0, #3
          bne     ENDCHECK
          mov     r0, r3
ENDCHECK

```

a) Indique qual o tipo da estrutura de “decisão e controlo” utilizada no código?

b) Qual o “valor” do registo **R0** no final?

10) Tendo um vetor / *string* de valores de 8 bit, elabore um programa em *assembly* ARMv7 capaz de contar os dígitos (supondo código ASCII). A *string* tem formato tipo C (acaba com 0). ‘0’ e ‘9’ representam os valores respetivos caracteres.

AV+FSP+MC

11) Tendo por base o *assembly* ARMv7 e o emulador VisUAL responda às seguintes questões:

- a) Desenvolva uma rotina que determine o número total de bits com o valor “1” de número de 32 bit, passado como parâmetro. Os registos de entrada e saída devem ser definidos por si.

[illegible]

- b) Desenvolva uma rotina que determine o número total de bits com o valor “1” de um vetor de valores de 32 bits utilizando (invocando) a sub-rotina anteriormente definida. A dimensão do vetor fica ao seu critério.

This image shows a blank sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There is no handwriting or other markings on the paper.

--	--

PRÁTICA / ASSEMBLY / PLATAFORMA GNU

12) Considere o trecho de código seguinte (suponha que as equivalências / valores estão corretos):

Código:	
exit:	<pre> mov r7, #OS_EXIT mov r0, #NORMAL swi 0 </pre>

a) Indique qual o objetivo / o que faz o código.

b) O código *assembly* eventualmente colocado imediatamente a seguir ao trecho apresentado é executado? Porquê?

13) Novamente com base no *assembly* ARMv7 e um sistema com as ferramentas GNU:

a) Desenvolva uma rotina que determine o número de vezes que um valor / carater passado como parâmetro aparece numa *string* do tipo C (finalizada por 0). A interface com a rotina deve ser a seguinte: **R0** (endereço do início da *string*) e **R1** (valor / carater); retorno **R0** (número de ocorrências do carater na *string*).

<div style="border-bottom: 1px solid black; margin-bottom: 5px;"></div> <div style="border-bottom: 1px solid black; margin-bottom: 5px;"></div> <div style="border-bottom: 1px solid black;"></div>	<div style="border-bottom: 1px solid black; margin-bottom: 5px;"></div> <div style="border-bottom: 1px solid black; margin-bottom: 5px;"></div> <div style="border-bottom: 1px solid black;"></div>
---	---

[illegible][illegible]

<hr/> <hr/>

This image shows a single page of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There is no handwriting or other markings on the paper.