

Nome: _____

Número: _____

Notas Importantes:

Durante a prova pode consultar a bibliografia da disciplina (slides, livros, enunciados e material de apoio a trabalhos práticos). No entanto, não é permitido o uso de computadores ou qualquer outro dispositivo electrónico, com a exceção de um calculadora simples.

Este é um teste de escolha múltipla e deverá assinalar sem ambiguidades as respostas na tabela apresentada em baixo. Cada pergunta corretamente respondida vale cinco pontos. Cada resposta errada desconta dois pontos e cada pergunta não respondida vale zero pontos. Uma nota final abaixo de zero pontos é automaticamente arredondada para zero valores.

Respostas: (indicar de forma legível a resposta A, B, C ou D, debaixo do número da questão)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

1. Considerando o contexto da hierarquia de memória, diga qual das seguintes afirmações é **VERDADEIRA**:
- A. Não há vantagem em transferir da memória central para memória cache blocos de apenas 1 word (32 *bits*).
 - B. Os diferentes níveis numa hierarquia de memória *cache* tiram partido apenas da localidade espacial.
 - C. O tempo médio de acesso à memória diminui sempre que se aumenta o tamanho dos blocos.
 - D. Os blocos que são enviados para a memória *cache* são apagados da memória principal.

2. Indique qual das seguintes opções é **VERDADEIRA**, considerando o programa em C em baixo.

- A. É apresentado no ecrã o valor 16.
- B. É apresentado no ecrã o valor 32.
- C. É apresentado no ecrã o valor 8.
- D. O programa termina com um erro do tipo *Segmentation Fault*.

```
#include <stdio.h>
int main() {
    int a[] = {1, 2, 4, 8, 16, 32, 64};
    int *p1, *p2, **p3;
    p1 = a+5;
    p2 = p1-3;
    p3 = &p1;
    printf("%d\n", *p2+*(p1-1)-*(*p3-3));
    return 0;
}
```

3. Considere a instrução *lw* em *Assembly* do MIPS. Das seguintes afirmações, diga qual representa uma afirmação **VERDADEIRA**.
- A. Sendo uma instrução de armazenamento de dados na memória, a instrução não faz uso da Unidade Aritmética e Lógica (ALU).
 - B. A instrução está activa em todas as etapas do *dapapath*.
 - C. Não sendo uma instrução TAL, esta instrução é sempre decomposta e, portanto, executada em mais do que um ciclo.
 - D. Todas as restantes afirmações são falsas.

4. Considere que tem uma memória *cache* com um tamanho total de **512 kB**, organizada em blocos de **256 bytes**, numa arquitectura de 32 *bits* em que o endereçamento é efectuado ao nível do *byte*. Sabendo que a estrutura de endereço da memória se decompõe num campo *Tag* de **16 bits**, num campo *Index* de **8 bits** e um campo *Offset* de **8 bits**, estamos perante uma memória *cache* de que tipo?

- A. 2-way set-associative cache.
- B. 8-way set-associative cache.
- C. Direct mapped cache.
- D. 4-way set-associative cache.

5. A codificação em código máquina da instrução **beq** incluída no excerto de código *Assembly* do MIPS listado a seguir é:

- A. 0x1192FFFA
- B. 0x1192FFF6
- C. 0x1192FFF7
- D. 0x1192FFF8

```
Loop:    andi $16,$0,0x000FFFFF
         addi $17,$17,1
         addi $18,$16,0x1234
         slti $8,$17,5
         mul $12,$11,100
         beq  $12,$18,Loop
```

6. Diga qual das seguintes afirmações é **VERDADEIRA**:

- A. As chamadas a funções podem ser feitas utilizando instruções do tipo *branch*.
- B. Uma instrução *j* (*jump*) permite apenas fazer saltos para zonas de memória de instruções próximas da posição actual do PC.
- C. O deslocamento relativo de uma instrução *branch* permite codificar saltos entre -16384 e +16383 instruções.
- D. A instrução de salto que permite maior liberdade para saltar para qualquer zona de memória é a instrução *jr*

7. Indique em que conjunto de instruções *Assembly* do MIPS é que se decompõe a instrução
`ori $t0,$t1,0xF0E12D3C`:

- A. `lui $at, 0xF0E1`
`andi $at, $at, 0x2D3C`
`or $t0, $t1, $at`
- B. `lui $at, 0xF0E1`
`ori $at, $at, 0x2D3C`
`or $t0, $t1, $at`
- C. `lui $at, 0x2D3C`
`ori $at, $at, 0xF0E1`
`and $t0, $t1, $at`
- D. `lui $at, 0x2D3C`
`ori $at, $at, 0xF0E1`
`or $t0, $t1, $at`

8. Assumindo que a *label* **array** se refere a uma tabela de inteiros armazenada no endereço de memória **0x00080000**, e que as *labels* **func1** e **func2** são referências externas ao ficheiro, indique quantas entradas na tabela de realocação gerará o seguinte código *Assembly* do MIPS?

- A. 3 entradas na tabela de realocação.
- B. 5 entradas na tabela de realocação.
- C. 4 entradas na tabela de realocação.
- D. 6 entradas na tabela de realocação.

```
loop:
    la $a0,array
    lw $t0, 0($a0)
    lw $t1, 4($a0)
    addu $t2,$t1,$t0
    sw $t2, 8($a0)
    blt $t0,100, loop
    la $a0, array
    lw $t3, 24($a0)
    jal func1
    move $v0, $0
    lw $ra, 16($sp)
    addiu $sp,$sp,32
    jal func2
```

9. Considere a instrução em *Assembly* do MIPS `and $a0,$a2,$a1`. Sabendo que os registos \$a0, \$a1 e \$a2 são os registos **#4**, **#5** e **#6**, respectivamente, indique qual dos seguintes códigos hexadecimais corresponde à codificação desta instrução:

- A. 0x00C52024
- B. 0x00A62024
- C. 0x00A43024
- D. 0x00C52020

10. Indique qual é o valor armazenado no registo **\$t0** após a execução do excerto de código *Assembly* do MIPS listado abaixo:

- A. 0x00112200
- B. 0x1122FFFF
- C. 0x00000000
- D. 0x00223300

```
addi $t0,$0,0x11223344
sll  $t0,$t0,8
ori  $t0,$t0,0x000000FF
srl  $t0,$t0,16
sll  $t0,$t0,8
andi $t0,$t0, 0xFF0000FF
```

11. Considere um sistema baseado num processador com um valor de **CPI REAL** igual a **5.0**. O sistema possui duas *caches*, uma para instruções e outra para dados. A *hit rate* da cache de instruções é de 90% e a *miss rate* da *cache* de dados é igual a 30%. A *miss penalty* em ambas as memórias é igual a **10** ciclos de relógio. Sabendo que apenas 40% das instruções envolvem um acesso à memória de dados indique qual o **CPI IDEAL** deste sistema?

- A. 3.8 B. 2.8 C. 5.0 D. 2.2

12. No desenvolvimento de um programa e teste com *debugger*, qual das seguintes sequências de comandos é válida. Note que o ponto e vírgula permite separar dois comandos consecutivos:

- A. `gdb -g f1.c f2.s -o main; gcc main`
 B. `gcc -g -c f1.c f2.s -o main; gdb main`
 C. `gcc -c f1.c f2.s -O main; gdc main`
 D. `gcc -g f1.c f2.s -o main; gdb main`

13. Assumindo que, num programa em *Assembly* do MIPS, dispõe de um número inteiro **com sinal** num registo **\$s0** e que se pretende calcular a sua divisão por 16, indique qual das seguintes instruções poderá ser utilizada para esse propósito.

- A. `srl $s0, $s0, 4` C. `sll $s0, $s0, 4`
 B. `sra $s0, $s0, 4` D. Nenhuma das restantes opções.

14. Suponha que tem uma memória composta por dois níveis de *cache* (**L1** e **L2**, respectivamente) e a memória principal. Assuma os seguintes dados relativos a essa memória:

- **L1 cache**: *hit time* de 10ps;
- **L2 cache**: *hit rate* de 80% e um *hit time* duas vezes superior ao da cache de nível 1;
- Memória principal: *hit time* de 200ps;

Indique qual deverá ser a *hit rate* para a memória *cache* **L1**, para que o valor do tempo médio de acesso a dados em memória seja de 16ps.

- A. 90% C. 3.50%
 B. 10% D. 80%

15. Dos seguintes segmentos de código em C, indique o que é equivalente ao ciclo em *assembly* listado a seguir:

- A) `while (($s3>=$s4) && ($s5>=$s4)) {s0++;}`
 B) `while (($s3>=$s4) || ($s5>=$s4)) {s0++;}`
 C) `while (($s3<$s4) && ($s4>=$s5)) {s0++;}`
 D) `while (($s3<$s4) || ($s5<$s4)) {s0++;}`

```

loop:
    slt    $t0,$s3,$s4
    slt    $t1,$s5,$s4
    add    $t0,$t0,$t1
    beq    $t0,$0,out
    addi   $s0,$s0,1
    j      loop
out:
  
```

16. Indique qual das seguintes afirmações é **FALSA**:

- A. A instrução **jr** (*jump register*) é uma instrução do tipo **R**.
 B. A instrução **beq** (*branch if equal*) é uma instrução do tipo **I** em que o campo *immediate* especifica um deslocamento relativo no espaço de endereços.
 C. O tamanho do campo *shift amount* na codificação de uma instrução do tipo **R** é 5 *bits* o que quer dizer que o deslocamento máximo possível de realizar com uma instrução de *shift* aritmético é **31**.
 D. Como o tamanho do campo *immediate* das instruções do tipo **I** é de 16 *bits*, isto significa que é possível com instruções *branch* dar saltos relativos ao *program counter* de $\pm 2^{16}$ *words*.

17. Considerando o trecho de programa indicado ao lado, indique qual das seguintes afirmações é **VERDADEIRA**:

- A. A variável **Data** vai ser armazenada na zona de dados estáticos do programa, a variável **tab** e **k** na pilha. A variável **tab** vai conter um endereço localizado na *heap*.
- B. As variáveis **Data**, **k** e **tab** vão ser armazenadas na pilha, sendo que **tab** aponta para uma zona de memória na *heap*.
- C. A variável **Data** vai ser armazenada na zona de dados estáticos, a variável **k** na pilha e a variável **tab** é armazenada na *heap*.
- D. As variáveis **k** e **tab** vão ser armazenadas na *heap* porque são variáveis locais.

```
int Data;

void main() {
    int k,*tab;

    tab=(int*)malloc(10*sizeof(int));

    for(k=0;k<10;k++)
        ...
}
```

18. Indique qual das seguintes afirmações é **VERDADEIRA**:

- A. O mecanismo de *polling* é mais eficiente do ponto de vista de utilização do processador do que o mecanismo baseado em interrupções.
- B. Num processador RISC, que se preocupe em simplificar o hardware, vamos encontrar sempre uma interface de comunicação com dispositivos externos do tipo I/O programado.
- C. O mecanismo ideal para transferir grandes quantidades de informação entre um dispositivo de entrada/saída e o processador/memória é o acesso directo à memória.
- D. O mecanismo baseado em interrupções é mais fácil de gerir/programar do que o mecanismo de *polling*.

19. Considere o excerto de código ao lado em *Assembly* do MIPS. Indique que valores irão ficar armazenados nos registos \$t1 e \$t2.

- A. O registo **\$t1** irá ficar com o valor **-24** e o registo **\$t2** com o valor **-12**.
- B. O registo **\$t1** irá ficar com o valor **-48** e o registo **\$t2** com o valor **-24**.
- C. O registo **\$t1** irá ficar com o valor **-48** e o registo **\$t2** com o valor **-6**.
- D. O registo **\$t1** irá ficar com o valor **48** e o registo **\$t2** com o valor **-24**.

```
...
addi $t0,$0,-12
sll $t1,$t0,2
sra $t2,$t1,1
...
```

20. No *Assembly* do MIPS suponha que pretende realizar a multiplicação de dois números armazenados nos registos \$a1 e \$a2. Se quiser usar apenas instruções TAL, escolha qual dos seguintes excertos de código permitiria colocar o resultado da multiplicação no registo \$s0.

- | | |
|--------------------------------|-------------------------------|
| A) mult \$a1,\$a2
mflo \$s0 | C) mult \$s0,\$a1,\$a2 |
| B) mult \$a1,\$a2
mfhi \$s0 | D) mul \$a2,\$a1
mflo \$s0 |