T1:____
T2:____
T3:____
T4:____
T5:____
T6:____
T7:____
T: ____
P:_____

NOME: _____ **Nº:**_____

TEÓRICA

As questões devem ser respondidas na própria folha do enunciado. As questões 1 a 4 são de escolha múltipla, e apenas uma das respostas está correcta, valendo 1 valor. Uma resposta errada desconta 1/3 de valor. As questões 5 e 6 valem 2 valores cada. A questão 7 (4 valores) só deve ser respondida pelos alunos que, justificadamente, não fizeram a componente teórico-prática.

1. Considere a instrução `leal %eax, 1000( %ebx, %edx, 2)`. Que valor é colocado no registo `%eax` sabendo que `%ebx=1000`, `%edx=10`, a palavra de memória com endereço 1000 tem o valor 150 e a palavra com endereço 2020 tem o valor 60?

☐ 2020
☐ 60
☐ 150
☐ 210

2. Considere o seguinte código em C:

```
struct {char a; char b} vector[1000];
int i=10, x;
x = vector[i].b;  /***/
```

Qual das seguintes sequências de instruções *assembly* do MIPS implementa a instrução C assinalada com asteriscos, sabendo que `$t0` está associado a `x`, `$t1` contem o endereço base de `vector` e `$t2` contem o valor de `i`?

☐ `sll $t3, $t2, 2`
`add $t1, $t1, $t3`
`lb $t0, 1($t1)`
	☐ `add $t1, $t1, $t2`
	`lb $t0, 1($t1)`

☐ `add $t3, $t2, $t2`
`add $t1, $t1, $t3`
`lb $t0, 1($t1)`
	☐ `add $t3, $t2, $t2`
	`add $t1, $t1, $t3`
	`sb $t0, 1($t1)`

3. Sabendo que Texec = #I * CPI * Tcc, podemos dizer que um aumento para o dobro na frequência do relógio do processador resulta:

☐ Num Texec maior pois maiores frequências correspondem geralmente a maiores tempos de execução.
☐ Em diminuir Texec para metade pois o ciclo do relógio (Tcc) passa para metade.
☐ Num ganho em tempo de execução inferior ao aumento da frequência, devido a atrasos associados a outros componentes, tais como os barramentos e memória.
☐ Num ganho em tempo de execução inferior ao aumento da frequência, devido a um aumento no número de instruções.

4. Relativamente ao *data path* de ciclo único do MIPS podemos dizer que:

☐ É ineficiente pois todas as instruções necessitam do mesmo período de tempo para executar.
☐ Permite obter o melhor desempenho pois o CPI de qualquer instrução é 1.
☐ É eficiente pois cada instrução apenas necessita dos períodos de tempo associados às várias fases da sua execução.
☐ É eficiente pois cada componente apenas é usado uma vez para a execução de cada instrução.

NOME: _____ **Nº:**_____

5. Dependendo do nível de optimização pretendido os compiladores utilizam vários mecanismos de optimização do código, entre os quais se incluem: redução do número de acessos à memória, cálculo mais eficiente de endereços e expressões, utilização de instruções ou sequências de instruções mais eficientes e movimento de código (code motion). Diga o que entende por cada um destes mecanismos, ilustrando com exemplos se necessário.

NOME: _____ **Nº:**_____

6. Considere o data path de ciclo único do MIPS apresentado em anexo. Caracterize as várias fases de execução da instrução "beq", o valor dos sinais de controlo (com excepção de ALUop), bem como o momento em que estes são inicializados.
   Porque razão são necessárias 2 componentes diferentes de memória, um para dados, outro para instruções?

NOME: _____ Nº:_____

Apenas para os alunos que, justificadamente, não fizeram a componente teórico-prática.

7. Na década de 80 assiste-se ao aparecimento de uma filosofia de desenho de processadores designada genericamente pelo nome "Reduced Instruction Set Computer (RISC)". Os seus defensores proponham que, comparativamente com as arquitecturas CISC:
   - fosse reduzido o número de instruções e estas realizassem apenas operações elementares
   - as instruções tivessem tamanho fixo e o número de diferentes formatos de instruções fosse reduzido
   - o número de modos de endereçamento fosse reduzido
   - o número de registos visíveis ao programador fosse alargado.

Apresente argumentos para cada uma destas propostas.

NOME: _____ **Nº:**_____

PRÁTICA

As questões práticas devem ser respondidas em folha separada. A questão 1 vale 4 valores, as questões 2 e 3 valem 2 valores cada.

1. Considere o programa escrito em linguagem C

P1a:___
P1b:___
P1c:___
P1d:___
Σ: ____

```
int vect[200];
main () {
  int x;
  x = busca(vect);
  printf("%d",x);
}
int busca(int v[200]) {
  int i, ret=0;
  for (i=0; i<200; i++)
    if (v[i]>5)
      ret+=v[i];
    else
      ret-=v[i]/4;
  return(ret);
}
```

(a) Complete o programa em baixo, nas 6 zonas marcadas a sublinhado, de modo a fazê-lo corresponder a uma compilação válida do programa em C.

```
.LC0: .string    "%d"
main: pushl %ebp
      movl  %esp, %ebp
      pushl _________
      call  busca
      popl  %edx
      pushl %eax
      pushl $.LC0
      call  printf
      leave
      ret
busca:pushl     %ebp
      movl  %esp, %ebp
      xorl  %eax, %eax
      pushl %ebx
.L8:  xorl  %ecx, %ecx
      movl  _________, %ebx
.L10: movl  _____________, %edx
      cmpl  $5, %edx
      jle   _______
      addl  %edx, %eax
.L5:  incl  %ecx
      cmpl  $199, %ecx
      jle   ________
      popl  %ebx
.L6:  leave
      ret
.L7:  sarl  _______, %edx
      subl  %edx, %eax
      jmp   .L5
```

(b) Identifique e explique as instruções responsáveis pelo cálculo de "`x = busca(vect)`".

NOME: _____ **Nº:**_____

(c) Identifique e explique as instruções responsáveis pelo cálculo de "`ret-=v[i]/4`".

(d) Que alterações seriam necessárias fazer, no programa assembly, caso a instrução "`ret-=v[i]/4`", no programa C, fosse substituída por "`ret-=v[i]*15`"?

NOME: _____ **Nº:**_____

2. Considere o seguinte fragmento de código, escrito em assembly do MIPS:

P2a:___
P2b:___
P2c:___
P3a:___
P3b:___
Σ: ____

```
      ori  $s0,   $zero, 10
LB2:  sub  $t1,   $a2,   $s0   # QB
      sb   $a0,   9($t1)
      move $v0,   $t1
      subu $t1,   $t1,   $t2
      ble  $t1,   $zero, LB2
```

a) Reescreva o código fornecido, usando apenas instruções nativas do MIPS.

b) Converta a instrução assinalada com "**# QB**" para código máquina do MIPS, apresentando o resultado final em hexadecimal e incluindo na resposta todos os passos intermédios usados.

c) Converta para assembly do MIPS a instrução máquina **0x 28ABFFFA** (em hexadecimal), apresentando os cálculos intermédios.

NOME: _____ **Nº:**_____

3. Considere um computador com endereços de 64 bits e palavras também de 64 bits. Se a capacidade da memória cache for 512K e as suas linhas forem de 16 palavras, qual deve ser a capacidade total dessa cache (incluindo dados, tags e valid bits), quando a sua organização for:

a) Mapeamento completamente associativo?

b) Mapeamento directo?