não usar	
1	
2	
3	
4	
T	

1. Considere o seguinte excerto de um programa escrito em *assembly* e a executar numa máquina com cache:

```
ciclo:  movl 0(%ebx), %edx
        movl $10, 0(%ebx)
        addl $4, %ebx
        cmpl $0, %edx
        jnz ciclo
```

Considere que o registo `%ebx` aponta para o início de um array de inteiros (4 bytes) com os seguintes valores: -10, 30, 1024, -33, 0. Note que o ciclo termina quando o valor lido do array for 0. A frequência do relógio é de 2 GHz, o CPI_{CPU} é 2, a *miss rate* de instruções é de 4% e a de dados de 5%. Sabendo que o tempo de execução deste programa é de 150 ns, qual é a miss penalty (expressa em tempo)?

2. Indique se a afirmação abaixo é verdadeira ou falsa, justificando.
"A técnica de *pipelining* acelera o desempenho de um processador por permitir que uma única instrução processe vários elementos de dados. Trata-se de explorar um tipo de paralelismo conhecido como Single Instruction Multiple Data".

Nome: _____ Número:_____

3. A tabela abaixo apresenta na coluna da esquerda uma sequência de endereços (m=4) de acesso à memória gerados por um determinado programa. As 3 colunas seguintes referem-se a um modo de mapeamento numa cache que usa o algoritmo de substituição LRU. Preencha-as indicando em que set/linha (dentro do set) mapeia cada endereço, qual a tag associada a essa linha depois deste acesso e indicando se se trata de um cold miss, colisão ou de um hit. Considere a cache inicialmente fria.

Addr	**(S=2,E=2,B=2,m=4)**	**tag**	**cold miss/hit/colisão**
1			
13			
0			
6			
8			

4. Diga o que entende por dependências de dados no contexto de uma máquina com *pipeline*, de que forma estas podem prejudicar o desempenho do *pipeline* e que estratégia(s) conhece para minimizar este impacto.