C1	
1	
2	
3	
4	
T	
C2	
5	
6	
F	

COMPETÊNCIAS FUNDAMENTAIS

1. Considere o seguinte excerto de um programa escrito em *assembly* do Y86:

```
ciclo:  addl %edx, %eax
        subl %esi, %ecx       # %esi contem o valor 1
        jnz ciclo             # repete até %ecx==0
```

Sabendo que o valor inicial de %ecx é 1000, a frequência do relógio é de 1 GHz e o tempo de execução deste programa é de 4,5 µs (microsegundos), qual o CPI médio?

Texec = 4.5*10^-6 <=>
<=> 4.5*10^-6 = #I * CPI / f
<=> 4.5*10^-6 = (1000*3) * CPI / (1*10^9)
<=> CPI = 1.5 (cc/i)

2. Diga o que entende, justificando por palavras suas, pelo hiato processador-memória.

Hiato processador-memória:

"A memória é incapaz de alimentar o processador com instruções e dados a uma taxa suficiente para o manter constantemente ocupado"

Nome: _____ Número:_____

3. Considere o seguinte programa em *assembly* do Y86. Apresente o mesmo programa em código máquina, indicando os endereços de memória onde as instruções ou elementos de dados são colocados (utilize a base hexadecimal sempre que apropriado).

```
.pos 0x000
  jmp main
.pos 0x00F0A0
main:
  irmovl $10, %eax
  irmovl $0, %ebx
  mrmovl 0x0A0(%ebx), %ecx
  addl %eax, %ecx
  rmmovl %ecx, 0x0A0(%ebx)
  halt
```

0x000	jmp main	70 A0 F0 00 00	
0x00F0A0	irmovl $10, %eax	30 80 0A 00 00 00	
0x00F0A6	irmovl $0, %ebx	30 83 00 00 00 00	
0x00F0AC	mrmovl 0x0A0(%ebx), %ecx	50 31 A0 00 00 00	-> Atenção que nesta o rB é o EBX, não o ECX
0x00F0B2	addl %eax, %ecx	60 01	
0x00F0B4	rmmovl %ecx, 0x0A0(%ebx)	40 31 A0 00 00 00	
0x00F0BA	halt	10	

4. Indique, para a instrução `mrmovl 0x0A0(%ebx), %ecx` do programa anterior, o valor de todos os sinais relevantes da arquitectura SEQ do Y86 (suponha que a posição de memória lida tem o valor 0 (zero)).

Fetch		Memory	
Decode		Write Back	
Execute		PC	

Nome: _____ Número:_____

COMPETÊNCIAS C2

5. A organização SEQ do Y86 permite, sem qualquer modificação ao *datapath*, suportar instruções de retorno condicional, `retXX`. Estas instruções retornam de uma função de forma idêntica ao `ret`, se a condição XX (as mesmas condições que as suportadas para os saltos condicionais e determinada pelo campo *ifun*) for verdadeira, ou continuam na instrução seguinte se a condição for falsa. Indique na tabela abaixo a sequência dos sinais para uma ocorrência genérica desta instrução.

Fetch	icode : ifun valP = PC + 1	Memory	valM = M_4(valA)
Decode	valA = R[%esp] valB = R[%esp]	Write Back	R[%esp] = valE
Execute	valE = valB + 4 Bch = f(ifun,cc)	PC	PC = Bch? valM ; valP

6. Considere uma máquina com um espaço de endereçamento de 64 bits, com uma cache com capacidade para dados de 512 KBytes, linhas de 8 palavras e palavras de 8 bytes. Quantos bits são necessários para a tag se:

   a. O mapeamento for do tipo 8-*way set associative*?

   b. O mapeamento [redacted] full associative?

a)
De b) temos que a cache tem 8 K linhas. Essas 8K linhas distribuem-se em 8K/8 = 1K sets, o que faz com que o indice ocupe log2 (1K) = log2 (2^10) = 10 bits.
Logo a tag ocupa (64 -10 -3 -3 = 48 bits).

b)
Cada palavra tem 8 bytes, e cada linha 8 palavras, logo cada linha tem 64 bytes.
A cache tem (512 Kbytes = 2^9 * 2^10).
Logo há (2^9 * 2^10 / 2^6) = (2^3*2^10) = (8 K linhas). (log 2 (8K) = 13 bits) que ocupa o indice;
como são 8 palavras por linha precisamos de 3 bits para o block offset
e mais 3 para escolher o byte dentro da palavra.
Logo a tag ocupa (64 -13 -3 - 3 = 45 bits).

Nome: _____ Número:_____