Lista para a prova 1

1) Represente os números abaixo nas bases 2, 3, 8, 10 e 16.
    a. $(33)_{10}$
    b. $(54)_{8}$
    c. $(16)_{10}$
    d. $(F3)_{16}$
    e. $(87)_{10}$
    f. $(22)_{3}$
    g. $(122)_{10}$
2) Qual é o valor resultante em hexadecimal das constantes
    a. #(BIT8)
    b. #(BIT3|BIT4)
    c. #(BIT1|BIT0)
    d. #(BITF|BIT5)
    e. #(BITF|BITE)
    f. #(BIT9|BIT4|BIT0)
    g. #(BIT9|BIT8|BIT7)
3) Em assembly, como é possível modificar apenas 1 bit de um registro sem modificar os outros bits do mesmo registro? Dê exemplos para os casos de *setar* o bit, *limpar* o bit ou *inverter* seu valor.
4) É possível usar todos os 16 registros da CPU como se fossem de uso genérico?
    a. Descreva em poucas palavras a função de cada registro especial.
5) Qual é o tamanho mínimo e máximo (em palavras de 16-bits) das instruções de formato I (dois operandos). Dê exemplos de instruções que ocupam o tamanho mínimo e o tamanho máximo.
6) Repita a questão 5 para instruções de formato II (um operando)
7) Qual será o conteúdo dos bytes de memória nos endereços [0x2400 - 0x2405] depois de executada a instrução MOV.W #0x2345, *data*; se *data* está na posição 0x2403 da memória. Justifique.
8) Qual é o código de máquina gerado a partir das instruções abaixo?
    a. MOV.B R5, R8
    b. ADD.W var, 6(R5) ;[var = 0x2400] [end. desta instrução 0x4412]
    c. ADD.W &var, 6(R5) ;[var = 0x2400] [end. desta instrução 0x4412]
    d. RRL 12(R10)
    e. CMP @R12+,12(R6)
    f. JMP $
9) Desmonte as instruções abaixo. Se forem usados os geradores de constante R2 ou R3, interprete a instrução desmontada.

a)	b)
8077	5325
0030	4536
9077	8356
000A	90F5
3802	F0F0
12B0	0005
404E	37FB
5706	3FFF

10) Uma data de calendário pode ser representada pelo conjunto de 2 palavras de 8-bits, representado o dia e mês e uma palavra de 16-bits representando o ano. [DD-MM-AAAA]. Ao total, uma data ocupa 32-bits na memória. Escreva uma subrotina que compare datas e retorna R10 = 1 se data1 > data2, R10 = 0 se data1 = data2 ou R10 = -1 se data1 < data2. Considere passagem de parâmetro por endereços fixos na memória: data1 e data2.
11) Escreva uma rotina MUL que recebe dois operandos e realiza a multiplicação dos dois. Considere que os operandos são:
    a. Números de 8-bits sem sinal (MUL8U)
    b. Números de 16-bits sem sinal (MUL16U)
    c. Números de 8-bits com sinal (MUL8S)
    d. Números de 16-bits com sinal (MUL16S)
12) Escreva uma rotina converta todas as letras de uma string em letras minúsculas. Ex: “User's Guide” -> “user's guide”
13) Escreva a rotina DIV8 que recebe dois operandos em R4 e R5 de 16 bits e realiza a divisão R4/R5. Você deve fornecer o resultado da divisão e o resto em dois registros separados: R6 e R7.
14) Considere dois números complexos $a = R5 + i \cdot R6$ e $b = R7 + i \cdot R8$. Escreva um programa MULT8X que realize a multiplicação desses dois números. Você pode usar as subrotinas de multiplicação da questão 11.
15) Escreva uma subrotina utilizando a convenção de passagem de parâmetros pela pilha que conta quantos bits de uma palavra (16-bits) estão setados. Ex:

```
PUSH.W      #0x4400          ; 0x4400 tem apenas 2 bits setados
CALL        #CONTAUMS        ; Chama a subrotina
POP         R5               ; R5 deve conter o resultado = 2
```

16) Escreva uma subrotina utilizando a convenção de passagem de parâmetros pela pilha que apaga de uma string, todas as letras indicadas pelo registro R5. Ex:

```
MOV.B       #0x41,R5         ; R5 = ‘A’
PUSH.W      #string          ; 1º arg = string
PUSH.B      R5               ; 2º arg = letra a apagar do vetor
CALL        #APAGALETRA      ; Chama a subrotina
JMP         $                ; Resultado: string = “RR ZUL”
            .data
string      .byte       “ARARA AZUL”, 0x00
```

17) Como configurar a direcionalidade dos pinos?
18) Explique a dupla funcionalidade do registro PxOUT
19) Configure os pinos P1.0 e P4.7 como saídas.
20) Configure os pinos P1.1 e P2.1 como entradas com resistor de pull-up.
21) Configure os pinos P3.[0-7] como entradas com resistor de pull-down em apenas 3 linhas de código.
22) Repita o ex. 21 para os pinos P5.3 e P5.7
23) Escreva uma rotina que acende o LED de P1.0 enquanto o usuário estiver com o botão de P2.1 pressionado
24) Escreva uma rotina que muda o estado do LED verde toda vez que o usuário pressionar e soltar S2.