Linguagem Assembly – Ficha Nº2

1. O que são registos?

2. Num computador, onde existem os registos?

3. Para que servem os registos?

4. Qual a relação entre os registos de propósito geral de 8 e 16 bits? Exemplifique.

5. Considerando os seguintes dados:

   Registos: AX = 30F5H
   BX = FF05H
   CX = 2A02H
   DX = 1FEBH

   Data Segment:

ENDEREÇO	CONTEÚDO
246AH	50H
246BH	FFH
246CH	2CH
FF04H	ACH
FF05H	0DH
FF06H	10H

   Descreva o funcionamento e o resultado de cada uma das seguintes instruções:

   a) MOV AX, BX
   b) MOV CL, 37H

c) MOV CX, [246BH]

d) MOV CX, 246BH

e) MOV AX, [BX]

f) ADD AL, DH

g) MUL BX

h) DIV BL

i) SUB AX, DX

j) OR CL, BL

k) NOT AH

l) AND AL, 0FH

6. Considerando os seguintes dados:

Registos: AX = 30F5H
BX = FF05H
CX = 2AE5H
DI = FFF0H

Data Segment:

ENDEREÇO	CONTEÚDO
0003H	00H
0004H	FFH
0005H	22H

Explique o funcionamento e o resultado dos seguintes grupos de instruções:

a) ADD BL, AL
MOV [0004], BL

b) MOV BX, 0004H
MOV AL, [BX]
SUB AL, CL
INC BX
MOV [BX], AL

7. Dado o seguinte programa em Assembly indique:

    a) As variáveis.
    b) O espaço em bytes ocupado por cada variável.
    c) O valor de iniciação de cada variável, em decimal.
    d) O significado da instrução mov al, first.
    e) O valor do registo AX nas linha 15, 16 e 17.
    f) A utilidade da instrução da linha 12.
    g) A utilidade das instruções das linhas 13 e 14.
    h) Seria possível substituir a instruções das linhas 13 e 14 por uma outra equivalente? Porquê?
    i) O significado da instrução cmp al, ah.
    j) Uma instrução equivalente a jna.
    k) O que faz o programa.

```
1: .8086
2: .model small
3: .stack 2048
4:
5: DATA_HERE   SEGMENT
6:        first     DB   25H
7:        second    DB   31H
8:        greater   DB   ?
9: DATA_HERE   ENDS
10:
11:CODE_HERE   SEGMENT
12:       ASSUME CS:CODE_HERE, DS:DATA_HERE
13:       mov ax, DATA_HERE
14:       mov ds, ax
15:       xor ax,ax
16:       mov al, first
17:       mov ah second
18:       cmp al, ah
19:       jna sec
20:       mov greater, al
21:       jmp acabou
22:  sec: mov greater, ah
23:  acabou:mov ah, 4ch
24:       int 21h
25: CODE_HERE  ENDS
26: END
```

8. Faça um programa que some dois números de 8 bits, cujos endereços iniciais são Address1 e Address2, deixando o resultado num espaço reservado a partir de Address3. Identifique cada uma das instruções.

9. Faça um programa que copie uma cadeia de caracteres de um vector de bytes para outro. Os endereços iniciais dos vectores são respectivamente *Vector1* e *Vector2*. A cadeia de caracteres é terminada por um caracter de valor zero.

10. Faça um programa que some os elementos de dois vectores colocando o resultado num 3º.

11. Faça um programa que construa um novo vector, a partir de um vector já existente, cujos valores dos elementos sejam o dobro dos valores originais. O número de elementos do vector é dado pela constante *NumElem*. O endereço do vector já existente é *OldVector*, e o do vector a criar é *NewVector*.

12. Faça um programa que permita contar todos os espaços em branco de uma cadeia de caracteres em memória. A cadeia de caracteres é guardada como um conjunto de células terminadas pelo valor zero e de início no endereço START.

13. Elabore um programa que permita contar os caracteres minúsculos, maiúsculos e dígitos numéricos de uma cadeia de caracteres em memória. A cadeia de caracteres é guardada como um conjunto de células terminadas pelo carácter $.

14. Faça um programa que converta todos os caracteres maiúsculos de uma string, terminada pelo caracter zero, para minúsculos.

15. Faça um programa em Assembly que some os valores ímpares presentes num vector de bytes. O vector terá no máximo 20 elementos. Caso tenha menos elementos, o último valor será seguido do valor 0.

16. Considerando que a variável NUMERO contém um valor numérico compreendido entre [0 - 65535], faça um programa, em linguagem *Assembly* que conte o número de dígitos a zero existentes em NUMERO. Suponha que o número se encontra na base 10. O resultado deverá ficar armazenado na variável QUANT. (Ex: Para NUMERO=10230, QUANT=2).

17. Faça um programa em *Assembly* que transforme uma sequência de caracteres, cada um representando um valor inteiro de um único dígito, num único valor numérico em que cada um dos seus dígitos são os elementos referidos. Assuma que o primeiro elemento da cadeia de caracteres tem um valor indicativo do número total de elementos que a constitui. Assuma também que o valor gerado não é superior a 65535.

    Exemplo: String 5, '1','2', '3', '4', '8'
    Número gerado 12348