Assembly Language - Facha nº2

1. What are records?

2. On a computer, where are the records?

3. What are the records for?

4. What is the relationship between 8 and 16 -bit general purpose records?

   Exemplify.

5. Considering the following data:

   Records: Ax = 30f5h bx = ff05
   h cx = 2a02h dx = 1Febh

Data Segment:

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

Describe the operation and result of each of the following instructions:

a) MOV AX, BX

b) MOV CL, 37h c) M
OV CX, [246BH] D) M
OV CX, 246BH E) MO
V AX, [BX] F) ADD A
L, DH G)

6. Considering the following data:

Records: Ax = 30f5h bx = ff05
h cx = 2ae5h di = fff0h

Data Segment: Address Content 0003h 00h 0
004h ffh 0005h 22h

Explain the operation and result of the following instruction groups:

a) add bl, al mov [000
4], BL B) MOV BX, 0
004H MOV AL, [BX]
SUB AL, CL INC BX
MOV [BX], AL

7. Given the following meeting in Assembly Indicate:

a) The variables. b) the bytes space occupied by each variable. c) the initiation value of each variable, in decimal. d) The meaning of the MOV AL, First. e) T he value of the AX registration in line 15, 16 and 17. f) the usefulness of the inst ruction of line 12. g) the usefulness of lines of lines 13 and 14. h) Would it be p ossible to replace the instructions on lines 13 and 14 with another equivalent? Why? i) The meaning of the CMP al, ah. j) An instruction equivalent to JNA. k) What the program does.

1: .8086 2: .Model Small 3: .Sstack 2048 4: 5: Date_here Segmen t 6: First DB 25H 7: Second DB 31h 8: Greater DB? 9: Date_Her e Ends 10: 11: Code_Here Segment 12: Assumes Cs: Code_Here, DS: Data_Here 13: MOV AX, Date_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: JMP 22: SEC: MOV GREATER , AH 23: It's over: Mov ah, 4ch 24: int 21h 25: code_here ends 26 : end

8. Make a program that adds two numbers of 8 bits, whose initial addresses are Address1 and Address2, leaving the result in a space reserved from Address3. Identify each of the i nstructions.

9. Make a program that copies a character chain from one bytes vector to another. The i nitial addresses of the vectors are respectively *Vector1* and *Vector2*. The character chain is terminated by a zero value character.

10. Make a program that adds the elements of two vectors by placing the result in a 3rd.

11. Make a program that builds a new vector, from an existing vector, whose elements v alues are twice the original values. The number of elements of the vector is given by the constant *NumElem*. The existing vector address is *OldVector*, and the vector to create is *NewVector*.

12. Make a program that allows you to tell all the blank spaces of a character chain in memory. The character chain is stored as a set of cells ending by the zero and sta rt value at Start address.

13. Develop a program that allows you to count the tiny, capital and numerical digi ts of a character chain in memory. The character chain is stored as a set of cells en ding by the character $.

14. Make a program that converts all the capital characters of a string, finished by t he zero character, for tiny.

15. Make an assembly program that adds the odd values present in a bytes vector. The vector will have a maximum of 20 elements. If you have fewer elements, the l ast value will be followed by the value 0.

16. Considering that the variable number contains a numerical value between [0 - 65535] , make a program, in language *Assembly* that counts the number of zero digits existing in number. Suppose the number is in base 10. The result should be stored in the Quant vari able. (Ex: For number = 10230, quant = 2).

17. Make a program into *Assembly* that transforms a sequence of characters, each repres enting an integer value of a single digit, in a single numerical value in which each of its d igits are the elements referred to. Assume that the first element of the character chain has a value indicative of the total number of elements that constitutes it. Also assume that the value generated is not greater than 65535.

Example: string	5, ‘1′, ′,‘ 3 ′,‘ 4 ′,‘ 8 ′
Generated number	12348