I

architecture. How do we obtain the correspon [illegible] from its concrete representation in base 2? Which are in this context the possible methods/techniques/mechanisms to be applied ? Detail all these mechanisms through a proper discussion and analysis on an adequate representation example (ex: discussion and analysis on the representation of -37 OR -912 etc.). How many and which are the specific cases in which it is necessary to apply the mechanisms described above and how exactly are they applied? (description + 1 concrete example for each of the mentioned cases). What do we mean by and what is the difference between the notions of "representation" and "interpretation" at the level of the 80x86 architecture? Explain and exemplify. What does "admissible representation interval" mean and which are those for the most frequently used values at a program level? Explain and justify how is the value 0 (zero) considered and why.

b). Operands specification modes. Give one adequate and suggestive source code example for each specification mode. Explain and justify. What do you understand by direct addressing vs. indirect addressing ? Exemplify and explain.

II. Present the corresponding memory layout of the following segment (its structure and corresponding memory values, assuming that its starting offset is 0), justifying in context the reason for obtaining each of the stored values:

```
segment data
x dw -130, 100+100b+100h
y dw 1100h>>1001b, 100h-1
z dw z-$, $-x+y-z
w dw x, y, z, w
h dw 101b,101-h,11h-11b,h-11

a db $-$$
  dw $$-a
b dd a+b-0bh+2,a-b+0ah-0bh
c db 3-b, b-3
d dw 513, -513
e dd abcdefh, 'abcdefh'

f dw b-1, [b-1]
g times 3 dd 'db'
k dw 12345678h>>4<<4, b+0bh
m dd a+0ah, a+ah
s dd a-start, start-start1
;start and start1 def in code segment
```

If there are data elements or source lines which you consider to be syntactically incorrect, motivate and explain yo decision and ignore further that values or lines in providing the requested memory layout. That is, if ALL elements o line are syntactically incorrect, the ENTIRE line is removed/ignored, if only SOME elements are syntactically incorr ONLY THOSE elements are removed/ignored, the other syntactically correct elements remaining valid and they hav be reflected in the segment data structure as an effect.

III. a) Answer each of the questions below accurately, justifying the answers given in the context with the nec explanations that have to justify, clarify and detail the mechanisms involved.

1). Which is the MINIMUM number of bits needed to represent the numbers **64** and **-257**, respectively? Justify. G values in both base 2 and base 16.

2). The instruction **„mov dword [esp+4], [esp]”** provides unfortunately "syntax error". Why ?. Write a seque **correct** assembly language instructions in place of the given one to be equivalent as effect to the intention su the given instruction. Explain and justify.

). Is the instr. **movsx ax,[6+esp]** correct? If so, explain what it does and which is its effect. If not, justify the
). Give 2 examples of different asm instructions for which both implicit operands to be of different sizes. Ex
. Give 2 examples of different asm instructions for which both implicit operands to be from memory . Exp

jmp [fs:edx + edi * 4 - 56] and **[jmp FAR [fs:edx + edi ***

Which is the difference between the instructions **jmp [fs:edx + edi * 4 - 56]** and **[jmp FAR [fs:edx + edi *** he operands size of these instructions and which is the exact effect of each one ? In how many ways ca jump at the level of the 80x86 architecture in 32 bits programming ? Why ? How about a NEAR jum erence ? Explain.

Vrite an instruction that sets SF, CF and OF simultaneously to value 0. Explain.

A matrix having the dimensions NxM is a two-dimensional table containing data which can be repre
ctor of NxM elements. The following 4x3 matrix:

4	15	3
1	8	12
14	17	6
3	2	5

can be linearized in a doubleword vector in the following manner: *matrix dd 4, 15, 3, 1, 8, 12, 14, 17, 6, 3, 2,*
and columns being being numbered starting from 1. In this case, the element with the value 15 (from line 1
of the matrix above corresponds to the second element, and the element with the value 6 (from line 3,
corresponds with the ninth element from the *matrix* vector.

1. Write in assembly language a subprogram named ***get_element*** which receives 4 parameters: the addr
doubleword array which represents a linearized matrix and three positive numbers represented on dou
M (the number of columns of the matrix), **I** (the current line) and **J** (the current column). The subprog
provide as output the value of the element from the matrix specified by these 4 parameters.

2. Two numbers **M** and ***N*** having values lower than 10 and a ***matrix*** array of doublewords are given in
segment. The program will display on the screen the matrix associated with the given array in two-dim
format (that is, line by line and column by column), using for this purpose the ***get_element*** subprogram.

3. Read from the keyboard a doubleword **X.** Verify if the given number is an element of the matrix, in wh
the value **X** will be printed, together with the number of its occurrences and the positions from the
which **X** appears. Otherwise, the message *„Element does not exist!"* will be printed.

**Explain and detail the algorithm approach and the mechanisms involved. Justify and comment accordingly th
code. It is not mandatory to implement a multi-module solution.**

<u>Example:</u>

M db 3
N db 4
matrix dd 4, 15, 3, 1, 8, 12, 14, 17, 6, 3, 2, 5

It will display the matrix in the format given above. For the read value X=3 the message ***„3 appears 2 times in***
(1,3); (4,1)" *will be displayed on the screen, and for X=7 the message* ***„Element does not exist!"*** *will be printed.*

Working time: 2h 45' Evaluation: I – 2.50p (1.75p+0.75p); II – 1.5p ; III – 2p; IV – 3p ; (+1p – by def