I. Efectuați o analiză asupra conceptului de „depășire” la nivelul operațiilor aritmetic scoțând apoi în evidență cum reacționează arhitectura 80x86 la apariția unei astfel de situații în cazul fiecăruie dintre cele 4 tipuri de operații aritmetice de bază: adunare, scădere, înmulțire și împărțire. Care sunt situațiile considerate de către limbajul de asamblare „depășiri” și cum reacționează procesorul? Prezentați pentru fiecare din cele 4 cazuri câte 1-2 exemple de cod sursă adecvate și sugestive pentru apariția unor astfel de situații, explicați care este reacția sistemului de calcul în fiecare caz și ce poate face programatorul ulterior apariției unei astfel de situații la nivelul programului.

II. Se dă un șir de valori numerice întregi reprezentate pe dublucuvinte. Să se tipărească pe ecran stringul pozițiilor dublucuvintelor din șirul de dublucuvinte care au cel de-al doilea octet al reprezentării în baza 16 strict negativ, precum și suma acestor octeți în baza 10. Explicați algoritmul, justificați și comentați corespunzător codul sursă. Insistați în explicații asupra aspectelor dificile sau problematice implicate în soluția prezentată.

Ex: Se dă șirul de dublucuvinte: sir dd 12A63478h, 12345678h, 1A2B3C4Dh, FEDC9876h
Cel de-al doilea octet al reprezentării în baza 16 este: A6h, 34h, 2Bh și respectiv DCh. Valorile lor numerice corespunzătoare sunt: -90, 52, 43 și respectiv -36, deci dublucuvintele ce au cel de-al doilea octet al reprezentării în baza 16 strict negativ sunt pe pozițiile 1 și 4 în șirul dat, așadar trebuie afișat pe ecran stringul „14”, precum și suma celor 2 octeți în baza 10 (-126).

IV. Definiți și explicați semnificația și importanța noțiunilor de cod de apel, cod de intrare și cod de ieșire în contextul general al limbajelor de programare. Analizați cum se reflectă acestea la nivelul limbajului de asamblare 80x86 și respectiv la nivelul sitvei de execuție în cadrul rulării unui program.

Timp de lucru: 2h 30'
Punctaj: I - 2p; II - 3p; III - 2p; IV - 2p; (+1p din oficiu)

III. Explicați și justificați efectul fiecărei linii sursă (în măsura în care o considerați corectă sintactic - dacă nu, explicați de ce este incorectă și treceți mai departe...), arătând cum funcționează exact fiecare instrucțiune și care va fi conținutul regiștrilor implicați, în fiecare dintre bazele de numerați 2, 10 și 16 (și în interpretarea cu semn și în cea fără semn).

```
x db -128,128, -1
y dd -128
z dw 128
...
mov ax,word ptr x+1
not ah
neg al
mov bx, word ptr x+2
xchg bh,bl
neg bx
cbw
xor ax,bx
xor bx,ax
sub ax,bx
les di,y+1
inc di
push ds
pop es
mov ch,es:[di]
inc di
mov dx,es:[di]

mov ax,-129

cbw
and ax,-1
cwd
shl ax,2
sar al,2
or ax,dx
div al
```

Timp de lucru: 2h 30'
Punctaj: I - 2p; II - 3p; III - 2p; IV - 2p; (+1p din oficiu)

.Problema: Erau definite niste dwords. Unul din bytes de la fiecare word trebuia pus intr-un string si daca era negativ trebuia afisat in binar