Exercice 1 *(12.5 Pts)* *(Temps recommandé : 30 mn)*

On donne les valeurs du registre index suivant : SI = 1000H ;
On considère également le programme ASM 80x86 suivant :

```
5000 : 0500          MOV CX , n ;          (1)
5000 : 0502   BCL :  MOV [SI], BX ;        (2)
5000 : 0505          MOV BX, [DI] ;        (3)
5000 : 0508   BCL :  ADD SI , 2 ;          (4)
5000 : 050A          ADD DI , 2 ;          (5)
5000 : 050C          DEC CX  ;             (6)
5000 : 050E          (instruction manquante) ;  (7)
5000 : 0511          END ;                 (8)
```

1. (n) étant un entier naturel, quel sera sa valeur (*exprimée en HEXADECIMAL*) si l'on veut transférer un bloc de 40 octets ? *(1pt)* *Taille (BX)=2 octets => n = 40/2= 20 = 14h .*
2. Corriger les lignes (2) et (3) pour que l'on puisse transférer des données depuis [SI] vers [DI]. *(1pt)* *(2) -> MOV BX, [SI] ; et (3) -> MOV [DI], BX ;*
3. Trouver l'instruction manquante en ligne (7). Justifier votre réponse *(1pt)* *« JNZ BCL »*
4. Indiquer la taille de l'instruction (7). Justifier votre réponse *(1pt)* *511-5OE=3 octets*
5. Le programme ci-dessus comporte une erreur générant une *erreur de fonctionnement* : trouver cette erreur et proposer une correction. Justifier votre réponse *(1.5pt)* *BCL en (4) à enlever.*
6. On remplace, dans les instructions (2) et (3), le registre (BX) par le registre (BL); Quelle sera la nouvelle valeur (en HEXADECIMAL) du nombre (n) ? *(2pts)* *Taille(BL)=1 octet => n = 40/1= 40 = 28h .*
7. On suppose une pile avec un pointeur (SP) à sa valeur maximale : SP = SPmax ; cette pile est-elle *complètement vide* ou *complètement pleine* ? (justifier) *(2pts)* *Complètement VIDE*
8. On remplace l'instruction (3) : « **MOV [DI], BL** ; » par l'instruction « **PUSH BL ;** » ; Quelle sera la valeur finale de SP (**exprimée en hexadécimal**) en fin d'exécution de ce programme ? (on suppose que le segment de pile (SS) est un segment de 64Ko) *(3pts)* $SP_{final}=SP_{max} - 40=FFFF-28\ (h) = FFD7\ h$

Exercice 2 *(7.5 Pts)* *(Temps recommandé : 30 mn)*

Rédiger un programme en ASSEMBLEUR 80x86 qui calcule la somme des 50 premiers nombres supérieurs à « 5 », saisis depuis une zone RAM commençant à l'adresse pointée par (SI) (SI quelconque).

Rem : 1- Un nombre **A** est supérieur à « 5 » si « **A**-5 » est positif ;
2- Un nombre **B** est négatif si son MSB (*Most Significant Bit*) est = 1.

```
                MOV CX , 50 ;       Compteur " 50 nbres nuls" ds. CX
                MOV DX , 0 ;        Résultat (somme) dans DX
SAISI :         MOV AX , [SI] ;     Saisi ..
                MOV BX, AX ;         .. puis .. Sauvegarde ..
                SUB BX, 5 ;
                AND BX , 8000h ;          ..puis masque pour extraire le
                                    bit (b15) : signe
                JNZ INFERIEUR ;           si (b15≠0/cas  inférieur à « 5 »),
                                    alors saut INFERIEUR
                ADD DX, AX ;        sinon(b15=0 /cas inférieur à « 5 »),
                                    MAJ (Somme dans DX)..
                JMP Prochain ;      .. et on ne fait pas d'incrément sur CX .
INFERIEUR : INC CX ;                .. cas (inf. à 5) : CX=CX+1(itération non
                                    ; comptabilisée)
Prochain :      ADD SI , 2 ;        prépare prochaine donnée à traiter
                LOOP BCL1 ;         Retour SAISI Tant que CX≠0
                End ;
```

Bon Courage