Bases des Systèmes Informatiques

Jeudi 8 janvier 2009 **Contrôle Assembleur**

Exercice 1

1. Commenter le code suivant.

```
-e200 10              ; écrit 10 à l'adresse [200]
-a100                 ; commence l'assemblage en [100]
      MOV AX,[200]    ; initialise AX à la valeur de [200] (AX=10)
      MOV BX,0        ; initialise BX à 0
      MOV CX,0        ; initialise CX à 0
LOOP: ADD BX,BX       ; ligne étiquetée LOOP, double BX (BX+=BX)
      ADD CX,BX       ; ajoute BX à CX (CX+=BX)
      ADD BX,1        ; incrémente BX de 1 (BX++)
      CMP BX,AX       ; compare BX à AX
      JLE LOOP        ; si BX<=[200] boucle à la ligne LOOP
      MOV [200],CX    ; sinon écrit valeur de CX en [200]
      INT 3           ; fin du programme
-p=100                ; exécute le programme débuté en [100]
-d200,201             ; affiche le contenu de [200] et [201]
```

2. Que fait ce programme ?

Il remplace la valeur 10 stockée en [200] par 22.

Exercice 2

On va écrire un programme qui ira chercher à l'adresse mémoire 200 une borne supérieure B et à l'adresse 201 un multiplicateur M. Pendant l'exécution du programme, tous les multiples de M strictement inférieurs à B devront être additionnés dans le registre AX. Par exemple, si M=2 et B=10 alors AX=2+4+6+8=20.

1. Ecrire un algorithme de calcul du résultat final n'utilisant que l'addition en pseudo-code, ou en C. On supposera que B et M sont initialisés.

```
while (M<B)
{
      X+=M;
      M+=M;
}
AX=X;
```

2. Traduire l'algorithme de la question 1 en assembleur.

```
      MOV AX,0
      MOV BX,[201]
LOOP: ADD AX,BX
```

```
        ADD BX,BX
        CMP BX,[200]
        JL  LOOP
        INT 3
```

Exercice 3

On souhaite écrire un programme effectuant le calcul de A élevé à la puissance P en ne se servant que de l'addition. On supposera, une fois de plus, que A est stocké à l'adresse 200 et P à l'adresse 201.

1. Comme dans l'exercice précédent, écrire un algorithme calculant le résultat.

```
A=[200];
P=[201];
B=A;
C=0;
for (i=1;i<P;i++)
{
      for (j=1;i<=A;j++)
      {
           C+=B;
      }
      B=C;
      C=0;
}
[202]=C;
```

2. Transcrire l'algorithme ci-dessus en assembleur et terminer le programme en écrivant le résultat de l'opération à l'adresse 202.

```
      MOV AX,[200]
      MOV BX,0
      MOV CX,1
      CMP CX,[201]
      JE  L3
 L1:  MOV DX,1
 L2:  ADD BX,AX
      ADD DX,1
      CMP DX,[200]
      JLE L2
      MOV AX,BX
      MOV BX,0
      ADD CX,1
      CMP CX,[201]
      JL  L1
 L3:  MOV [202],AX
      INT 3
```