Solution TD 4 : Architecture des ordinateurs
2020/2021

Question 1 **Le braille est une écriture en relief pour les malvoyants. Les caractères de cette écriture sont constitués de six points en relief (trou ou bosse) sur une grille de trois lignes et deux colonnes. Quelle est la *quantité d'information* d'un caractère de ce code ? Montrez que ce code suffit pour écrire pratiquement tous les textes.**

Réponse :

- **un point = 1 bit ; un caractère = 6 bits**
- **6 bits => 2^6 = 64 caractères. 26 lettres (pas forcé de différencier majuscules et minuscules) 10 chiffres, il reste encore de la marge pour la ponctuation.**

Question 2 : **Qu'est-ce qu'un mode d'adressage ? Quel registre particulier est utilisé pour l'adressage direct ?**

Réponse : Méthode utilisée pour interpréter l'opérande (absente, valeur, adresse…) En adressage direct, l'opérande est une adresse. Chargée par le bus d'adresse vers un Registre Tampon pour les Adresses ; Puis envoyée sur le bus d'adresse pour désigner une donnée en mémoire.

Question 3 **Parmi les instructions suivantes, indiquer celles qui sont incorrectes et corrigez-les**

Instruction	OK ?	Proposition de correction
PUSH AL	Non	PUSH AX
MOV AX, [1]	Oui	
ROL AX, 2	Non	ROL AX, 1
CMP [1000], 2	oui	
MOV AX, Tempo	Oui	
MOV AX, BL	Non	MOV AX, BX

Question 4 : manipulation de boucle : Ecrire un programme assembleur qui réalise l a somme des éléments d'une suite de nombre le résultat sera dans AX

La suite : pour n = 1 à10 , $A_n = A_{n-1} + 2 \times (A_{n-1} + r)$, $A_0 = 2$, r = 5

```
        mov ax,3
        mov bx,10
        mov cx,2
boucle: mul cx
        mov cx,ax
        add cx,10
        mov ax,3
        sub bx,1
        jz  fin
        jmp boucle
fin:    mov ax,cx
```

hlt

Question 5 Sous programme

Exemple : On va écrire une procédure SOMME qui calcule la somme de deux nombres naturels de 16 bits convenons que les entiers sont passés par les registres AX et BX et que le résultat sera placé dans le registre AX

La procédure s'écrit

```
SOMME    PROC  near
         ADD  AX, BX .  AX  ← AX + BX
         RET
SOMME    ENDP
```

On écrit

```
         MOV   AX, 6
         MOV   BX, 7
         CALL SOMME
         HLT
```

Ecrire un sous programme qui calcul le factoriel de n n ! = 1 x 2 x 3 x ……..(n-1) x n

n > 0 , 0 ! = 1, 1 ! = 1 ,

n ! = 1 x 2 x 3 x ……n

Remarque : Mul dx
multiplication de dx par ax

- Exemple : prendre n = 7 , 7 ! = $(5040)_{10}$ = $(13B0)_{16}$

Programme assembleur

factriel

```
      MOV AX,7
      MOV BX,6
l1:   MUL BX
      SUB BX,1
      JZ l2
      JMP l1
l2:   HLT
```

Sous Programme

factriel

```
        MOV AX,7
        MOV BX,6
        CALL  FACT
        HLT

FACT    PROC    NEAR
  l1:  MUL BX
       SUB BX,1
       JZ l2
       JMP l1
  l2:   ret
FACT  ENDP
```

Question 6 écrire un sous programme qui calcul Y^x

Y^x = Y x Y x Y x ….. (X fois) Exemple Y = 4 , X = 3,

Y^x = 4 x 4 x 4 (3 fois) 4^3 = $(64)_{10}$ = $(40)_{16}$

Programme assembleur

Puissance

```
    MOV AX,4
    MOV BX,AX
    MOV CX,2
l1: MUL BX
    SUB CX,1
    JZ l2
    JMP l1
l2: HLT
```

Sous Programme

Puissance

```
   MOV AX,4
    MOV BX,AX
    MOV CX,2
    CALL  puiss
   HLT
PUISS  PROC   NEAR
l1:  MUL BX
    SUB CX,1
    JZ l2
    JMP l1
l2:  RET
PUISS  ENDP
```