Exercice 1 (*2*0.5 + 2*1.5 = 4 pts*)

1- Effectuer les opérations arithmétiques suivantes :
(a) FFFEh + 1002h (11000h) ***(b)*** 1002h – FFFEh = 1002 + C_2(FFFE) = 1002 + (C_1(FFFE)+1) =1002 + (0001+1) = (1004h)

2- Simplifier l'expression booléenne suivante en utilisant la méthode formelle (algébrique) :

$$F = (a.b.c + a.\bar{b}.\bar{c}) + (\bar{a} + \bar{b}.c + b.\bar{c}).a \qquad (F = a)$$

3- Simplifier l'expression booléenne suivante en utilisant la table de Karnaugh :

$$G = x.(\overline{y \oplus z}) + (\bar{x} + (y \oplus z)).x \qquad (G = x)$$

Exercice 2 (*16 pts*)

On dispose de **20** données, de deux octets chacune, stockées en pile; on suppose que le pointeur **SP** n'a pas été modifié après leur empilement. On souhaite trier ces 20 données selon leur parité (paire/impaire) : les données paires seront transférées à l'adresse **(SI)** quelconque, les données impaires seront transférées à l'adresse **(SI+500)**. On donne aussi le programme ci-contre, permettant d'effectuer un tri :

```
                MOV BX , 0050h ; (1)
                MOV CX , 0032h ; (2)
                MOV AX , 00h ;  (3)
                JZ   FIN ;      (4)
                MOV CX , 0032h ; (5)
CHARGE :        MOV AX , [BX] ;(6)
                MOV  DX , AX ; (7)
                AND AX , 1 ;    (8)
                JZ  TYPE_1 ;    (9)
                JNZ  TYPE_2 ;  (10)
TYPE_1:         MOV [SI],  DX ;(11)
                ADD SI,  2 ;    (12)
                ADD BX,  2 ;    (13)
                LOOP  CHARGE ; (14)
                JMP  FIN ;      (15)
TYPE_2:         PUSH  DX ;      (16)
                ADD BX,  2 ;    (17)
                LOOP  CHARGE ; (18)
                JMP  FIN ;      (19)
FIN :           END ;           (20)
```

1- Ce programme comporte quatre (4) instructions inutiles et/ou générant une erreur de programme : Corriger le en les supprimant: *justifier votre réponse*. (*2 pts*) **Rép** : enlever ce qui est en *BLEU* : lignes (3, 4, 5, 19)

2- Indiquer la (ou les) instruction(s) responsables de la saisie des données à trier : *justifier votre réponse*. (*1 pt*) **Rép** : Ligne (6)

3- Indiquer quel type de tri est effectué dans ce programme (préciser le critère du tri) : *justifier votre réponse*. (*1 pt*) **Rép** : Paire/Impaire

4- Indiquer la caractéristique principale des données traitées dans la partie « TYPE_2 » de ce programme : *justifier votre réponse*. (*1 pt*) **Rép** : Impaire

5- Modifier ce programme pour l'adapter à la fonction souhaitée de tri des données issues de la pile. (*3 pts*) Indication : instructions concernées : (1, 2, 6, 13, 16, 17)
Ligne (1) : à enlever – **Ligne(2) :** MOV CX, 20 ; - **Ligne(6) :** POP AX ; - **Ligne(13) :** à enlever ; **Ligne (16) :** MOV [SI+500], DX ; **Ligne(17) :** à enlever.

6- Quelle est la valeur finale de (SP) si sa valeur avant exécution du programme est « 00AAh » ? (*2 pts*) SP_final = SP_Initial + 20*2 octets = 00AA + 28h = 00D2 h ;

7- Question **d'excellence** (*6 pts*): On souhaite restituer les données triées (maintenant réparties entre les blocs « SI » et « SI+500 ») dans la pile, en commençant par les données paires puis impaires. Proposer un programme assembleur permettant de réaliser cette tache
NB : on acceptera un organigramme ou un algorithme à défaut d'un programme assembleur. (*4 pts : Algorithme/organigramme **ou** 6 pts : programme ASSEMBLEUR*). Voir ci-dessous :

En fait, la solution en (5ème) est incomplète (mais admise) car il faudrait en toute rigueur utiliser un pointeur (DI) pour les données impaires, différent de (SI) utilisé pour les données paires, avec son initialisation en début de programme comme ci-dessous :

```
                MOV DI, SI                ; DI sera utilisé pour les données impaires
                MOV CX , 20 ; (2)
                MOV BX, 0          ;      BX est un compteur des nbres paires (init à 0)
CHARGE :        POP AX ; (6)
                MOV  DX , AX ; (7)
                AND AX , 1 ;   (8)
                JZ  TYPE_1 ;   (9)
                JNZ  TYPE_2 ;  (10)
TYPE_1:         MOV [SI],  DX ;(11)
                ADD SI,  2 ;     (12)
                INC BX ;                  MAJ du compteur de nbres paires
                LOOP  CHARGE ; (14)
                ~~JMP  FIN ;~~       (15)
                JMP  Répons_8 ;       (15)  On passe à la solution de Q8.
TYPE_2:         MOV [DI+500],  DX ;       (16)
                ADD DI,  2 ;
                LOOP  CHARGE ; (18)
Répons_8:       MOV CX, BX                ;CX = sauvegarde de BX : nbre de données
                                          ;paires (servira aussi de compteur)
REF_pair :      SUB SI, 2                 ; Pointe sur la dernière donnée paire
                MOV AX, [SI]              ; récupération dans AX
                PUSH AX                   ; sauve en pile
                LOOP REF_pair             ; retour à « REF_pair»tant que toutes les
                                          ; données n'ont pas été transférées.
                MOV AX, 20                ; le nbre total de données ds AX
                SUB AX, BX                ; AX = AX-BX=20-BX : nbre de données
                                          ; impaires
                MOV CX, AX                ; compteur des données impaires
REF_imp :       SUB DI, 2                 ; Pointe sur la dernière …
                MOV AX, [DI+500]          ;                  … donnée impaire
                PUSH AX                   ; sauve en pile
                LOOP REF_imp              ; retour à « REF_imp»tant que toutes les
                                          ; données n'ont pas été transférées.
FIN :           END ;          (20)
```

NB : cette solution n'est pas unique !