

Partiel S3 – Corrigé

Architecture des ordinateurs

Durée : 1 h 30

Exercice 1 (9 points)

Toutes les questions de cet exercice sont indépendantes. À l'exception des registres utilisés pour renvoyer une valeur de sortie, aucun registre de donnée ou d'adresse ne devra être modifié en sortie de vos sous-programmes. Une chaîne de caractères se termine toujours par un caractère nul (la valeur zéro). On dira qu'un caractère est blanc s'il s'agit d'un caractère *espace* ou d'un caractère *tabulation*.

- Réalisez le sous-programme **IsBlank** qui détermine si un caractère est blanc (c'est-à-dire s'il s'agit d'un espace ou d'une tabulation).

Entrée : **D1.B** contient le code ASCII du caractère à tester.

Sortie : Si le caractère est blanc, **D0.L** renvoie 0.
Si le caractère n'est pas blanc, **D0.L** renvoie 1.

Indication : La valeur numérique du code ASCII du caractère *tabulation* est 9.

```
IsBlank          ; Si le caractère est un espace, saute à blank.
                 cmpi.b #' ',d1
                 beq     \blank

                 ; Si le caractère est une tabulation, saute à blank.
                 cmpi.b #9,d1
                 beq     \blank

\not_blank       ; Le caractère n'est pas blanc, renvoie D0.L = 1.
                 moveq.l #1,d0
                 rts

\blank           ; Le caractère est blanc, renvoie D0.L = 0.
                 moveq.l #0,d0
                 rts
```

- Réalisez le sous-programme **BlankCount** qui renvoie le nombre de caractères blancs dans une chaîne de caractères. Pour savoir si un caractère est blanc, vous utiliserez le sous-programme **IsBlank**.

Entrée : **A0.L** pointe sur une chaîne de caractères.

Sortie : **D0.L** renvoie le nombre de caractères blancs de la chaîne.

Indications :

- Utilisez le registre **D2** comme compteur de caractères blancs (car **D0** est utilisé par **IsBlank**).
- Copier ensuite **D2** dans **D0** avant de sortir du sous-programme.

```

BlankCount      ; Sauvegarde les registres.
                 movem.l d1/d2/a0,-(a7)

                 ; Initialise le compteur de caractères blancs.
                 clr.l   d2

\loop            ; Charge un caractère de la chaîne dans D1.B.
                 ; Si le caractère est nul, on quite.
                 move.b  (a0)+,d1
                 beq     \quit

                 ; Si le caractère n'est pas blanc, on reboucle.
                 jsr     IsBlank
                 tst.l   d0
                 bne     \loop

                 ; Sinon, on incrémente le compteur de caractères.
                 addq.l  #1,d2
                 bra     \loop

\quit            ; Nombre de caractères blancs -> D0.L
                 move.l  d2,d0

                 ; Restaure les registres puis sortie.
                 movem.l (a7)+,d1/d2/a0
                 rts

```

3. Réalisez le sous-programme **BlankToUnderscore** qui convertit les caractères blancs d'une chaîne de caractères en caractères *underscore*. Pour savoir si un caractère est blanc, vous utiliserez le sous-programme **IsBlank**.

Entrée : **A0.L** pointe sur une chaîne de caractères.

Sortie : Les caractères blancs de la chaîne sont remplacés par des caractères « _ ».

```

BlankToUnderscore ; Sauvegarde les registres.
                  movem.l d0/d1/a0,-(a7)

\loop             ; Charge un caractère de la chaîne dans D1.B.
                  ; Si le caractère est nul, on quite.
                  move.b  (a0)+,d1
                  beq     \quit

                  ; Si le caractère n'est pas blanc, on reboucle.
                  jsr     IsBlank
                  tst.l   d0
                  bne     \loop

                  ; Sinon, on remplace le caractère blanc
                  ; par le caractère "underscore".
                  move.b  #'_',-1(a0)
                  bra     \loop

\quit             ; Restaure les registres puis sortie.
                  movem.l (a7)+,d0/d1/a0
                  rts

```

Exercice 2 (4 points)

Remplir le tableau présent sur le [document réponse](#). Donnez le nouveau contenu des registres (sauf le PC) et/ou de la mémoire modifiés par les instructions. **Vous utiliserez la représentation hexadécimale. La mémoire et les registres sont réinitialisés à chaque nouvelle instruction.**

Valeurs initiales : D0 = \$0004FFFD A0 = \$00005000 PC = \$00006000
 D1 = \$FFFF000A A1 = \$00005008
 D2 = \$FFFFFFFE A2 = \$00005010

\$005000 54 AF 18 B9 E7 21 48 C0
 \$005008 C9 10 11 C8 D4 36 1F 88
 \$005010 13 79 01 80 42 1A 2D 49

Exercice 3 (3 points)

Remplissez le tableau présent sur le [document réponse](#). Donnez le résultat des additions ainsi que le contenu des bits N, Z, V et C du registre d'état.

Exercice 4 (4 points)

Soit le programme ci-dessous :

```

Main      move.l   #44AA77FF,d7 ; 44AA77FF -> D7.L
next1     moveq.l  #1,d1         ; 00000001 -> D1.L
          tst.w    d7           ; Mise à jour de N et de Z en fonction de D7.W.
          bmi     next2         ; Saut si N = 1 (D7.W < 0).
          moveq.l  #2,d1         ; Sinon, 00000002 -> D1.L
next2     clr.l    d2           ; 00000000 -> D2.L
          move.w   #1234,d0      ; 1234 -> D0.W (D0.B = 34)
loop2     addq.l   #1,d2         ; D2.L + 1 -> D2.L
          subq.b   #1,d0         ; D0.B - 1 -> D0.B ; Seul D0.B est décrémenté.
          bne     loop2         ; Saut si Z = 0 (D0.B ≠ 0)
next3     clr.l    d3           ; 00000000 -> D3.L
          move.w   #1234,d0      ; 1234 -> D0.W
loop3     addq.l   #1,d3         ; D3.L + 1 -> D3.L
          dbra     d0,loop3      ; DBRA = DBF ; D0.W - 1 -> D0.W
          ; Saut si D0.W ≠ -1 (D0.W ≠ $FFFF)
next4     moveq.l  #1,d4         ; 00000001 -> D4.L
          cmp.b    #70,d7        ; Compare D7.B à la valeur 70.
          blt     quit          ; Saut si D7.B < 70 (comparaison signée).
          moveq.l  #2,d4         ; Sinon, 00000002 -> D4.L
quit      illegal

```

Complétez le tableau présent sur le [document réponse](#).

Nom : Prénom : Classe :

DOCUMENT RÉPONSE À RENDRE AVEC LA COPIE**Exercice 2**

Instruction	Mémoire	Registre
Exemple	\$005000 54 AF 00 40 E7 21 48 C0	A0 = \$00005004 A1 = \$0000500C
Exemple	\$005008 C9 10 11 C8 D4 36 FF 88	Aucun changement
MOVE.B -1(A2), -(A1)	\$005000 54 AF 18 B9 E7 21 48 88	A1 = \$00005007
MOVE.L \$500E, -1(A1,D0.W)	\$005000 54 AF 18 B9 1F 88 13 79	Aucun changement
MOVE.L #\$500E, -8(A0,D1.W)	\$005000 54 AF 00 00 50 0E 48 C0	Aucun changement
MOVE.W \$500A(PC), -(A1)	\$005000 54 AF 18 B9 E7 21 11 C8	A1 = \$00005006

Exercice 3

Opération	Taille (bits)	Résultat (hexadécimal)	N	Z	V	C
\$A3 + \$5C	8	\$FF	1	0	0	0
\$7005 + \$7005	16	\$E00A	1	0	1	0
\$7FFFFFFF + \$80000001	32	\$00000000	0	1	0	1

Exercice 4

Valeurs des registres après exécution du programme. Utilisez la représentation hexadécimale sur 32 bits.	
D1 = \$00000002	D3 = \$00001235
D2 = \$00000034	D4 = \$00000001