

T.P. 3 – Corrigé

La pile et les sous-programmes

Étape 4

```
LowerCount  movem.l  d1/a0,-(a7)
             clr.l   d0
\loop       move.b  (a0)+,d1
             beq    \quit
             cmp.b  #'a',d1
             blo    \loop
             cmp.b  #'z',d1
             bhi    \loop
             addq.l #1,d0
             bra    \loop
\quit       movem.l  (a7)+,d1/a0
             rts
```

Étape 5

```
UpperCount  movem.l  d1/a0,-(a7)
             clr.l   d0
\loop       move.b  (a0)+,d1
             beq    \quit
             cmp.b  #'A',d1
             blo    \loop
             cmp.b  #'Z',d1
             bhi    \loop
             addq.l #1,d0
             bra    \loop
\quit       movem.l  (a7)+,d1/a0
             rts
```

```

DigitCount  movem.l  d1/a0,-(a7)

            clr.l   d0

\loop      move.b  (a0)+,d1
            beq    \quit

            cmp.b  #'0',d1
            blo    \loop

            cmp.b  #'9',d1
            bhi    \loop

            addq.l #1,d0
            bra    \loop

\quit      movem.l  (a7)+,d1/a0
            rts

```

```

AlphaCount ; Compte le nombre de minuscules
           ; et empile le résultat.
           jsr    LowerCount
           move.l d0,-(a7)

           ; Compte le nombre de majuscules et l'additionne
           ; au sommet de la pile (sans dépiler).
           ; Sommet de la pile = Minuscules + Majuscules
           jsr    UpperCount
           add.l  d0,(a7)

           ; Compte le nombre de chiffres.
           ; Le sommet de la pile (Minuscules + Majuscules)
           ; est additionné au nombre de chiffres (D0).
           ; La somme est stockée dans D0.
           ; D0 = Minuscules + Majuscules + Chiffres
           ; Le sommet de la pile est dépilé (post incrémentation).
           jsr    DigitCount
           add.l  (a7)+,d0

           ; Retour de sous-programme.
           rts

```

Étape 6

```
Atoui      ; Sauvegarde les registres dans la pile.
           movem.l d1/a0,-(a7)

           ; Initialise la variable de retour à 0.
           clr.l   d0

           ; Initialise la variable de conversion à 0.
           clr.l   d1

\loop      ; On copie le caractère courant dans D1
           ; A0 pointe ensuite sur le caractère suivant (post incrémentation).
           move.b  (a0)+,d1

           ; Si le caractère copié est nul,
           ; on quitte (fin de chaîne).
           beq    \quit

           ; Sinon, on réalise la conversion numérique du caractère.
           subi.b  #'0',d1

           ; On décale la variable de retour vers la gauche (x10),
           ; puis on y ajoute la valeur numérique du caractère.
           mulu.w  #10,d0
           add.l   d1,d0

           ; Passage au caractère suivant.
           bra    \loop

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