# T.P. 1 – Corrigé
# Initiation

## Étape 7

```
GetMin              ; Comparaison entre D1.L et D2.L (D2 - D1).
                    ; L'instruction CMP ne modifie que les flags.
                    ; Les registres D1 et D2 ne sont pas modifiés.
                    cmp.l   d1,d2

                    ; Si D2 est inférieur ou égal à D1, on saute à d2min.
                    ; (LS = Low or Same, comparaison non signée.)
                    bls.s   d2min

d1min               ; D1 est inférieur à D2.
                    ; On le copie dans D0, puis sortie.
                    move.l  d1,d0
                    rts

d2min               ; D2 est inférieur à D1.
                    ; On le copie dans D0, puis sortie.
                    move.l  d2,d0
                    rts
```

## Étape 8

Voici une première solution possible pour **`StrLen`** :

```
StrLen              ; Sauvegarde A0 dans la pile.
                    move.l  a0,-(a7)

                    ; Initialise le compteur de caractères à 0
                    ; (D0 = compteur de caractères).
                    clr.l   d0

loop                ; Si fin de chaîne, on quitte.
                    ; (On profite également de ce test pour faire pointer A0
                    ;  sur le caractère suivant.)
                    tst.b   (a0)+
                    beq.s   quit

                    ; Sinon, on incrémente le compteur de caractères.
                    addq.l  #1,d0

                    ; Puis on reboucle.
                    bra.s   loop

quit                ; Restaure A0 puis sortie.
                    move.l  (a7)+,a0
                    rts
```

Voici une seconde solution que vous n'êtes pas obligés de comprendre pour le moment. Une optimisation, consistant à réduire au maximum les instructions de la boucle, a été réalisée. En effet, plus la chaîne est longue, plus le sous-programme passe de temps dans la boucle. Donc plus la boucle est courte, plus le sous-programme est rapide. C'est volontairement qu'aucun commentaire n'est présent dans le code : à vous de chercher.

```
StrLen              move.l  a0,d0

loop                tst.b   (a0)+
                    bne.s    loop

                    sub.l   a0,d0
                    add.l   d0,a0
                    not.l   d0

                    rts
```

## Étape 9

```
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
                    ; (puis on fait pointer A0 sur le caractère suivant).
                    move.b  (a0)+,d1

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

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