T.P. 4 – Corrigé
Calculatrice (partie 1)

Étape 1

```
RemoveSpace     ; Sauvegarde les registres dans la pile.
                movem.l d0/a0/a1,-(a7)

                ; Fait pointer A1 sur la chaîne destination.
                ; (Même chaîne que la source.)
                movea.l a0,a1

\loop           ; Charge un caractère de la chaîne dans D0 et incrémente A0.
                move.b  (a0)+,d0

                ; Si le caractère est un espace, saut à \loop.
                cmpi.b  #' ',d0
                beq     \loop

                ; Sinon, on copie le caractère dans la destination
                ; et on incrémente le pointeur destination.
                ; Si le caractère qui vient d'être copié n'est pas nul,
                ; saut à loop.
                move.b  d0,(a1)+
                bne     \loop

\quit           ; Restaure les registres puis sortie.
                movem.l (a7)+,d0/a0/a1
                rts
```

Étape 2

```
IsCharError     ; Sauvegarde les registres dans la pile.
                movem.l  d0/a0,-(a7)

\loop           ; Charge un caractère de la chaîne dans D0 et incrémente A0.
                ; Si le caractère est nul, on renvoie false (pas d'erreur).
                move.b   (a0)+,d0
                beq      \false

                ; Compare le caractère au caractère '0'.
                ; S'il est inférieur, on renvoie true (ce n'est pas un chiffre).
                cmpi.b  #'0',d0
                blo     \true

                ; Compare le caractère au caractère '9'.
                ; S'il est inférieur ou égal, on reboucle (c'est un chiffre).
                ; S'il est supérieur, on renvoie true (ce n'est pas un chiffre).
                cmpi.b  #'9',d0
                bls     \loop

\true           ; Sortie qui renvoie Z = 1 (erreur).
                ; (L'instruction BRA ne modifie pas le flag Z.)
                ori.b   #%00000100,ccr
                bra     \quit

\false          ; Sortie qui renvoie Z = 0 (aucune erreur).
                andi.b  #%11111011,ccr

\quit           ; Restaure les registres puis sortie.
                ; (Les instructions MOVEM et RTS ne modifient pas le flag Z.)
                movem.l (a7)+,d0/a0
                rts
```

Étape 3

```
IsMaxError      ; Sauvegarde les registres dans la pile.
                movem.l d0/a0,-(a7)

                ; On récupère la taille de la chaîne (dans D0).
                jsr     StrLen

                ; Si la chaîne a plus de 5 caractères, renvoie true (erreur).
                ; Si la chaîne a moins de 5 caractères, renvoie false (pas d'erreur).
                cmpi.l  #5,d0
                bhi     \true
                blo     \false

                ; Si la chaîne contient exactement 5 caractères :
                ; comparaisons successives avec '3', '2', '7', '6' et '7'.
                ; Si supérieur, on quitte en renvoyant une erreur.
                ; Si inférieur, on quitte sans renvoyer d'erreur.
                ; Si égal, on compare le caractère suivant.
                cmpi.b  #'3',(a0)+
                bhi     \true
                blo     \false

                cmpi.b  #'2',(a0)+
                bhi     \true
                blo     \false

                cmpi.b  #'7',(a0)+
                bhi     \true
                blo     \false

                cmpi.b  #'6',(a0)+
                bhi     \true
                blo     \false

                cmpi.b  #'7',(a0)
                bhi     \true

\false          ; Sortie qui renvoie Z = 0 (aucune erreur).
                ; (L'instruction BRA ne modifie pas le flag Z.)
                andi.b  #%11111011,ccr
                bra     \quit

\true           ; Sortie qui renvoie Z = 1 (erreur).
                ori.b   #%00000100,ccr

\quit           ; Restaure les registres puis sortie.
                ; (Les instructions MOVEM et RTS ne modifient pas le flag Z.)
                movem.l (a7)+,d0/a0
                rts
```

Étape 4

```
Convert         ; Si la chaîne est nulle,
                ; on quitte en renvoyant false (erreur).
                tst.b   (a0)
                beq     \false

                ; (À ce stade, la chaîne n'est pas nulle.)
                ; S'il existe une erreur sur les caractères,
                ; on quitte en renvoyant false (erreur).
                jsr     IsCharError
                beq     \false

                ; (À ce stade, la chaîne n'est pas nulle
                ;  et ne contient que des chiffres.)
                ; Si le nombre que contient la chaîne est supérieur à 32767,
                ; on quitte en renvoyant false (erreur).
                jsr     IsMaxError
                beq     \false

                ; La chaîne est valide, il ne reste plus qu'à la convertir
                ; puis à quitter en renvoyant true (aucune erreur).
                jsr     Atoui

\true           ; Sortie qui renvoie Z = 1 (aucune erreur).
                ori.b   #%00000100,ccr
                rts

\false          ; Sortie qui renvoie Z = 0 (erreur).
                andi.b  #%11111011,ccr
                rts
```

Étape 5

```
Print           ; Sauvegarde les registres dans la pile.
                movem.l d0/d1/a0,-(a7)

\loop           ; Charge un caractère de la chaîne dans D0.
                ; Si le caractère est nul, il s'agit de la fin de la chaîne.
                ; On peut sortir du sous-programme.
                move.b  (a0)+,d0
                beq     \quit

                ; Affiche le caractère.
                jsr     PrintChar

                ; Incrémente la colonne d'affichage du caractère,
                ; et reboucle.
                addq.b  #1,d1
                bra     \loop

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