T.P. 1 – Corrigé
Space Invaders (partie 4)

Étape 1

```
CopyLine            ; Sauvegarde les registres.
                    movem.l d0/d3/a1,-(a7)

                    ; Nombre d'itérations = Largeur en octets
                    ; Nombre d'itérations - 1 (car DBRA) -> D3.W
                    subq.w  #1,d3

\loop               ; Copie tous les octets de la ligne.
                    ; (Utilisation du OU pour rendre les pixels noirs transparents.)
                    move.b  (a0)+,d0
                    or.b    d0,(a1)+
                    dbra    d3,\loop

                    ; Restaure les registres puis sortie.
                    movem.l (a7)+,d0/d3/a1
                    rts
```

Étape 2

```
PixelToAddress      ; Sauvegarde les registres.
                    movem.l d1/d2,-(a7)

                    ; Détermine le nombre d'octets à ajouter en abscisse.
                    ; Divise l'abscisse par 8 :
                    ; Quotient -> D1.W (nombre d'octets)
                    ; Reste -> D0.W (décalage)
                    move.w  d1,d0
                    lsr.w   #3,d1
                    andi.w  #%111,d0

                    ; Détermine le nombre d'octets à ajouter en ordonnée.
                    ; Multiplie l'ordonnée par le nombre d'octets par ligne.
                    ; D2.W * BYTE_PER_LINE -> D2.L
                    mulu.w  #BYTE_PER_LINE,d2

                    ; Détermine l'adresse vidéo.
                    ; VIDEO_START + Nombre d'octets à ajouter en abscisse et en ordonnée.
                    lea     VIDEO_START,a1
                    adda.w  d1,a1
                    adda.l  d2,a1

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

Étape 3

```
CopyLine            ; Sauvegarde les registres.
                    movem.l d1-d4/a1,-(a7)

                    ; Nombre d'itérations = Largeur en octets
                    ; Nombre d'itérations - 1 (car DBRA) -> D3.W
                    subq.w  #1,d3

\loop               ; Octet à copier -> D1.B et D2.B
                    move.b  (a0)+,d1
                    move.b  d1,d2

                    ; Décale D1.B vers la droite de D0 bits.
                    lsr.b   d0,d1

                    ; Décale D2.B vers la gauche de (8 - D0) bits.
                    moveq.l #8,d4
                    sub.w   d0,d4
                    lsl.b   d4,d2

                    ; Copie D1.B et D2.B dans la mémoire vidéo.
                    or.b    d1,(a1)+
                    or.b    d2,(a1)

                    ; Reboucle tant qu'il y a des octets à copier.
                    dbra    d3,\loop

                    ; Restaure les registres puis sortie.
                    movem.l (a7)+,d1-d4/a1
                    rts
```

Étape 4

```
PrintBitmap         ; Sauvegarde les registres.
                    movem.l d0/a1,-(a7)

                    ; Adresse vidéo -> A1.L
                    ; Décalage -> D0.W
                    jsr     PixelToAddress

                    ; Copie la matrice de points du bitmap dans la mémoire vidéo.
                    jsr     CopyBitmap

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