TP CRAPS ARCHITECURE DES ORDINATEURS

```
// TP3 Exercice 1 (Calcul de la somme des entiers d'un tableau)
set T, %r1       //r1 = Tableau d'entiers
set n, %r2       //adresse de la longueur du tableau r1
ld [%r2], %r2    //r2 = la longueur du tableau r1
clr %r3          // r3 = i (parcours du tableau)
clr %r4          // r4 = S (somme du tableau)

loop: cmp %r3, %r2    //comparer i à n
  bgeu endloop        //arrêt si i >= n
  ld [%r1 + %r3], %r5 // r5 = T[i]
  add %r5, %r4, %r4   // r4 = r4 + T[i]
  add %r3, 1, %r3     // i = i + 1
  ba loop             // branchement à l'étiquette loop si la condition est vérifiée
endloop: ba endloop

T: .word 1,2,3,4,5,6,7,8,9,10
n: .word 10

//TP3 Exercice 2 (Calcul du max d'un tableau d'entiers)
set T, %r2           // Tableau T
set n, %r3           // r3 = adresse de la longueur du tableau T
ld [%r3], %r3        // r3 = longueur du tableau T
clr %r4              // (r4 = i) = 0

ld [%r2 + %r0], %r1   // max = T[0]
loop: cmp %r4, %r3    // comparer i à n
  bgeu endloop        // arrêt si i >= n
  ld [%r2 + %r4], %r5 // r5 = T[i]
  if: subcc %r1,%r5,%r0   // si r1 < T[i]
    bgeu end_if
    add %r5,%r0,%r1     // r1 = T[i]
  end_if:
```

```
  add %r4, 1, %r4     // i = i + 1
  ba loop             // branchement à l'étiquette loop si la condition est vérifiée
endloop: ba endloop

T: .word 1,5,2,9,10,4,3,6,7,8
n: .word 10

// TP3 Exercice 3 (Tri à Bulles d'un tableau d'entiers)
set T, %r2           // Tableau T
set n, %r3           // r3 = adresse de la longueur du tableau T
ld [%r3], %r3        // r3 = longueur du tableau T
add %r3, %r0, %r4    // (r4 = i) = n
sub %r4, 1, %r4      // (r4 = i) = n-1

loop_1: cmp %r4, 1       // comparer i à 1
  bl endloop_1           // arrêt si i < 1
  clr %r5                // (r5 = j) = 0
  sub %r4, 1, %r10
  loop_2: cmp %r5, %r10     // comparer j à i-1
    bg endloop_2            // arrêt si j >= i-1
    ld [%r2 + %r5], %r6     // r6 = T[j]
    add %r5, 1, %r7         // r7 = j + 1
    ld [%r2 + %r7], %r8     // r8 = T[j+1]
    if: cmp %r6, %r8        // si T[j] (r6) > T[j+1] (r8)
      ble end_if
      st %r6, [%r2 + %r7]      // T[j+1] = T[j] (r6)
      st %r8, [%r2 + %r5]      // T[j] (r6) = r9 (old T[j+1])
    end_if:
    add %r5, 1, %r5        // j = j + 1
    ba loop_2              // branchement à l'étiquette loop si la condition est
vérifiée
  endloop_2:
  sub %r4, 1, %r4          // i = i - 1
  ba loop_1                // branchement à l'étiquette loop si la condition est
vérifiée
```

```
endloop_1: ba endloop_1


T: .word 1,5,2,9,10,4,3,6,7,8
n: .word 10


// Version SP
// TP3 Exercice 3 (Tri à Bulles d'un tableau d'entiers)
set T, %r2           // Tableau T
set n, %r3           // r3 = adresse de la longueur du tableau T
ld [%r3], %r3        // r3 = longueur du tableau T
add %r3, %r0, %r4    // (r4 = i) = n
sub %r4, 1, %r4      // (r4 = i) = n-1


loop_1: cmp %r4, 1       // comparer i à 1
  bl endloop_1           // arrêt si i < 1
  clr %r5                // (r5 = j) = 0
  sub %r4, 1, %r10
  loop_2: cmp %r5, %r10     // comparer j à i-1
    bg endloop_2            // arrêt si j >= i-1
    ld [%r2 + %r5], %r6     // r6 = T[j]
    add %r5, 1, %r7         // r7 = j + 1
    ld [%r2 + %r7], %r8     // r8 = T[j+1]
    if: cmp %r6, %r8        // si T[j] (r6) > T[j+1] (r8)
      ble end_if
      call SP_Permuter
    end_if:
    add %r5, 1, %r5        // j = j + 1
    ba loop_2              // branchement à l'étiquette loop si la condition est
vérifiée
  endloop_2:
  sub %r4, 1, %r4          // i = i - 1
  ba loop_1                // branchement à l'étiquette loop si la condition est
vérifiée
endloop_1: ba endloop_1
```

```
T: .word 1,5,2,9,10,4,3,6,7,8
n: .word 10

SP_Permuter:
 st %r6, [%r2 + %r7]      // T[j+1] = T[j] (r6)
 st %r8, [%r2 + %r5]      // T[j] (r6) = r9 (old T[j+1])
ret

// Factorielle Récursive
n = 4
basepile=500
set basepile, %sp
set n, %r1
set 1, %r3

call SP_Fact_Rec
fin: ba fin



// n: r1, resultat: r3
SP_Fact_Rec:
  if: cmp %r1, 0
    beq null //si r1 = 0 sauter à null
    //sinon:
    deccc %r1 // r1 -= - 1
    push %r28
    call SP_Fact_Rec

    pop %r28
    inccc %r1   // r1 = r1 + 1
    umulcc %r3, %r1, %r3 // r3 = r3 * r1
    ba end_if
  null: set 1, %r3
  end_if:
  ret     //revient à pop %r28
```