TD. 1

1. Implémenter le Tri à Bulles

Corrigé :

```r
TriBulle <- function (ttab) {
  n <- length(ttab)

  for (i in n:2)
    for (j in 2:i)
      if ( (ttab[j-1] < ttab[j] ) )
      {
        temp <- ttab[j-1]
        ttab[j-1] <- ttab[j]
        ttab[j] <- temp
      }
  return(ttab)
}
#fin TriBulle

tab = c(12, 4, 0, 8, 6, 12, 22, 48, 16, 18, 0)

print(TriBulle(tab))
# ou bien si on veut sauver :
tabTrie <- TriBulle(tab)
tabTrie
# les donnees initiales sont inchangees
print(tab)
#----------
```

Faire exécuter, puis une séance de débogage pas à pas avec Rstudio en regardant l'évolution de tab (mettre un breakpoint en ligne 7 et exécuter les loops sur j). Est-il en place ? Stable ?

2. Implémenter le tri par insertion : TriInsertion

Corrigé:

```r
T1 <-Sys.time()

TriInsertion <- function(ttab) {
n <- length(ttab)

for (j in 2:n) {
  x <- ttab[j]
  i <- j-1
  while (( i > 0 ) && (ttab[i] > x)) {
    print(ttab[i])
    ttab[i+1] <- ttab[i]
    i <- i-1
  }
  ttab[i+1] <- x

} # fin for
return(ttab)
}  #fin TriInsertion

tab = c(12, 4, 8, 6, 12, 22, 48, 18, 0)
tabTrie <- TriInsertion(tab)
#----------

# temps de calcul
T2 <- Sys.time()
print(difftime(T2,T1))

# voir le resultat du tri
cat("tableau trie : ", tabTrie)
cat("tableau original", tab)
```

3. Voir le temps de calcul avec Sys.time() et difftime(T2,T1)
4. Pour obtenir un tri décroissant : tab[i] < x
5. Que se serait-il passé si la fonction travaillait sur tab ? (vérifier...)
   Noter que la fonction reçoit un paramètre non typé !

Corrigé :

```
T1 <-Sys.time()

TriInsertion <- function() {
n <- length(tab)

for (j in 2:n) {
  x <- tab[j]
  i <- j-1
  while (( i > 0 ) && (tab[i] > x)) {
    print(tab[i])
    tab[i+1] <- tab[i]
    i <- i-1
  }
  tab[i+1] <- x

} # fin for
return(tab)
}  #fin TriInsertion

tab = c(12, 4, 8, 6, 12, 22, 48, 18, 0)
tabTrie <- TriInsertion()
#----------

# temps de calcul
T2 <- Sys.time()
print(difftime(T2,T1))

# voir le resultat du tri
cat("tableau trie : ", tabTrie)
cat("tableau original", tab)
```

Dans cet ex. TriInsertion connait tab car se situe dans la mémoire globale et toute fonction du script peut y avoir accès. Par contre i, j et x sont des variables locales ! Mais TriInsertion ne modifie pas tab ! C'est un passage de paramètre par valeur. Les valeurs de tab sont utilisées mais la variable tab n'est pas modifiée. Si c'était possible on aurait un passage par variable (comme en C++ ou Pascal) . Et si la fonction recevait une variable et qu'elle soit appellée : TriInsertion(tab) ? Le résultat serait identique !

6. Ecrire un algorithme en pseudo-code qui accepte en entrée un tableau A de *n* éléments et un nombre *x* et qui renvoie un indice *i* tel que x =A[*i*] où *i* = -1 si *x* n'appartient pas à A. Calculer sa complexité.

Corrigé :

```
1  Fonction find(A,x) : i {
2  n = longueur(A); C1
3  i=1 ;            C2
4  Tant que ( (i < n+1) et (A[i] != x) ) C3*n
5  i = i + 1;           C4*n
6  finTantque
7  Si (i == i + 1)      C5
8        alors i = -1 ; C6
9  } // finFonction
```

T(n) = C1 + C2 + C5 + C6 + (C3 + C4)*n* = O(C + C'*n*) = O(*n*)