Algorithmique TD8:
Gestion des fichiers

1er décembre 2008

1 Manipulation des flots

1. Ecrire un algorithme qui recherche le nombre le plus proche d'un nombre x donné, dans un flot d'entier donné *tata*.

```
fonction recherche(tata:flot,x:entier):entier
  variables
    res: entier
    min:entier
  min ← MAXINT
  tant que non(eof(tata)) faire
    lire(tata,tmp)
    si abs(tmp-x) < min alors
      res ← tmp
      min ← abs(tmp-x)
    fsi
  fintantque
  retourner res
finfonction
```

2. Ecrire un algorithme qui lit au clavier une suite de 10 entiers, puis affiche ces nombres mais dans l'ordre inverse de leur introduction (sans utiliser de tableau).

```
procedure afficher(ind:entier)
 variables
   val:entier
 si ind > 0 alors
   lire(val)
   afficher(ind-1)
   ecrire(val)
 fsi
finprocedure
```

3. Ecrire un algorithme qui recherche les entiers n'apparaissant qu'une seule fois dans un fichier d'entiers.

```
procedure unefois(fic:chaine)
  variables
```

```
    f1:flot
    f2:flot
    val1:entier
    val2:entier
    nbfois:entier
  ouvrirl(f1,fic)
  ouvrirl(f2,fic)
  tant que non(eof(f1)) faire
    lire(f1,val1)
    nbfois ← 0
    debut(f2)
    tant que non(eof(f2)) et nbfois < 2 faire
      lire(f2,val2)
      si val2 = val1 alors
        nbfois ← nbfois+1
      fsi
    fintantque
    si nbfois < 2 alors
      ecrirenl(val1)
    fsi
  fintantque
  fermer(f1)
  fermer(f2)
finprocedure
```

2 Ecriture dans un fichier

Ecrire un algorithme qui permet d'écrire dans un fichier la saisie de l'entrée standard jusqu'à la saisie d'un mot clé passé en paramètre.

```
procedure saisir(cle:chaine)
  variables
    fichier:chaine
    val:chaine
    f:flot
  ecrire("entrer un nom de fichier")
  lirenl(fichier)
  ouvrire(f,fichier)
  ecrirenl("entrer vos valeurs")
  lire(val)
  tant que val ≠ cle faire
    ecrire(f,val)
    lire(val)
  fintantque
finprocedure
```

3 Lecture Ecriture dans un fichier

1. Ecrire une fonction qui permet de trier les lignes d'un fichier. Adapter la fonction de tris QuickSort.

```
fonction trifichier(fic:chaine)
```

```
  variables
    tab:tableau():chaine
    nbl:entier
    f:flot
    i:entier
  ouvrirle(f,fic)
  tantque non(eof(f)) faire
    getline(f)
    nbl ← nbl+1
  fintantque
  creertableau(tab,nbl)
  debut(f)
  pour i ← 1 a nbl pas 1
    tab(i) ← getline(f)
  fpour
  triRapide(t,1,nbl)
  debut(f)
  pour 1 ← 1 a nbl pas 1
    ecrire(f,tab(i))
  fpour
  fermer(f)
finprocedure
```

2. On va écrire une fonction qui permet de concaténer deux fichiers.
   (a) Ecrire une fonction qui prend deux fichiers en paramètres et ajoute le contenu du second à la fin du premier.

```
procedure concat(fic1:chaine,fic2:chaine)
  variables
    f1:flot
    f2:flot
    tmp:chaine
  ouvrire(f1,fic1)
  ouvrirl(f2,fic2)
  tantque non(eof(f1)) faire
    lire(f1,tmp)
  fintantque
  tantque non(eof(f2)) faire
    lire(f2,tmp)
    ecrire(f1,tmp)
  fintantque
  fermer(f1)
  fermer(f2)
finprocedure
```

   (b) Améliorer votre fonction en supprimant les doublons. Chaque ligne du second fichier n'est ajoutée au premier que si elle n'y est pas déjà.
   (c) Adapter votre fonction pour qu'elle utilise des fichiers déjà triés.