Lecture et écriture dans des fichiers

Tristan Colombo – 2003

Avant que de pourvoir lire dans un fichier, il faut l'ouvrir. Pour cela nous allons utiliser une variable d'un type particulier : FILE.

```
FILE *fic;
if ((fic=fopen("teste", "r"))==NULL)
{
      perror("fopen ");
      return(0);
}
```

Les lignes précédentes ouvrent le fichier « teste » en lecture (le « r » de la fonction fopen pour read … si,si …). La variable dédiée à ce fichier se nomme file. Pour ouvrir ce même fichier mais en mode écriture, il aurait fallu remplacer le « r » de fopen par un « w » (pour write).

Par la suite 2 possibilités :

- On a ouvert le fichier en lecture et on va donc y lire des données que l'on pourra stocker dans des variables. L'exemple suivant lit 4 variables par ligne (2 de type entier, et 2 de type caractère) jusqu'à la fin du fichier (fscanf renvoi EOF lorsque la fin du fichier est atteinte) :

  ```
  while (fscanf(fic, "%d %d %c %c\n", &i, &j, &c, &d)!=EOF)
        printf("%d %d %c %c\n", i, j, c, d);
  ```

- On a ouvert le fichier en écriture. On utilisera la fonction fprintf pour écrire dedans. Cette fonction se comporte comme un printf normal mais écrit dans le fichier spécifié :

  ```
  fprintf(fic, "1 2 T N\n");
  ```

On peut noter que la fonction printf est en fait un fprintf avec comme fichier de sortie stdout (l'écran), que le scanf correspond à fscanf avec comme fichier d'entrée stdin, et qu'il existe un troisième type de fichier spécial : stderr, le fichier de sortie d'erreur standard vers lequel on peut rediriger les messages d'erreur.

Lorsque toutes les opérations sont achevées sur le fichier, il faut penser à le refermer avec la commande fclose :

```
fclose(fic);
```

Voilà … vous pouvez donc manipuler des données sur des fichiers texte. Il faut savoir qu'il existe une autre façon de lire les fichiers en C en mode caractère par caractère et que certaines options permettent de traiter les fichiers binaires.