Introduction

Lire ou écrire un fichier à l'aide d'un langage de programmation c'est pratiquer des entrées/sorties.
Nous avons déjà pratiqué des entrées/sorties dans un environnement de type terminal et utilisant le *Shell*, l'interpréteur de commandes utilisé sur les systèmes *Unix*.
Par exemple, nous avons déjà récupéré les flux de sortie standard et d'erreur d'un programme (une commande) en utilisant des choses comme :

Exemple de récupération des flux de sorties

```
(ls fleurs.txt toto > sortie3) >& sortie4
```

Cela vous rappelle quelque chose ?

Les langages de programmation fournissent généralement un ensemble de fonctions permettant de lire ou écrire dans des fichiers présents sur le système de fichiers. *Le principe de fonctionnement d'un système de fichier sera illustré en cours.*

En langage C

En langage C, il est possible de lire le contenu d'un fichier en utilisant des fonctions disponibles dans la bibliothèque `stdio.h`.

Les éléments qui sont au programme et que vous devrez connaître sont l'utilisation des fonctions suivantes :

- **fopen** permet l'**ouverture** d'un flux correspondant à un fichier en vue de son utilisation en lecture ou en écriture, renvoie un **descripteur de fichier** ;
- **fclose** **fermeture** d'un flux associé à un fichier à partir du descripteur de fichier ⇒ opération à réaliser O-BLI-GA-TOI-RE-MENT ;
- **fscanf** lecture d'éléments dans une chaîne de caractères depuis un descripteur de fichier (noter le `f` en préfixe, qui permet de le distinguer de `scanf` qui lit depuis le flux d'entrée standard) ;
- **fprintf** écriture d'une chaîne de caractères dans le flux associé à un descripteur de fichiers.

L'écriture et la lecture dans un fichier se font de manière séquentielle. Les caractères sont lus les uns après les autres tant que le format correspond et que la fin du fichier n'est pas atteinte.
De la même manière, l'écriture s'effectue séquentiellement : les chaînes de caractères sont écrites les unes à la suite des autres. En pratique, l'écriture dans le flux associé à un fichier peut se faire via l'utilisation d'un *buffer*. Quoiqu'il en soit, le fichier de destination sur le disque n'aura le contenu attendu qu'une fois la fermeture du fichier effectuée.

Un exemple d'utilisation de ces fonctions est donné et expliqué sur la page suivante.

```c
// avec stdio.h, errno.h string.h assert.h

int main(int argc, char* argv[]) {
  assert (argc == 2);
  errno = 0;
  FILE* fd = fopen(argv[1], "r");
  if (fd == NULL) {
    printf("Erreur lors de l'ouverture du fichier %s : %d (%s)\n",
           argv[1], errno, strerror(errno));
    return 1;
  }

  int nb = 0;
  fscanf(fd, "%d", &nb);

  fclose(fd);
  return 0;
}
```

Explications du programme ci-dessus :

ligne 4 Vérification de la présence d'un argument sur la ligne de commande lors de l'appel du programme.

ligne 5 `errno` est une variable spéciale qui contient un code d'erreur lorsqu'un appel système ou certaines fonctions de bibliothèques échoue. Les codes obtenus peuvent permettre de déterminer l'erreur. La connaissance et l'utilisation de `errno` n'est pas au programme de la prépa, c'est toutefois très utile.

ligne 6 Ouverture du fichier dont le nom est donné en premier paramètre de `fopen`. Le deuxième paramètre est une chaîne de caractères qui détermine le mode d'ouverture. Les modes qui nous intéressent sont `"r"` pour la lecture (*read*) et `"w"` pour l'écriture (*write*). D'autres modes existent qui ne sont pas au programme.
La fonction `fopen` renvoie un pointeur de fichier (type `FILE*`), nommé ici `fd`.

ligne 7 Test de la validité du pointeur de fichier renvoyé. Ce test est O-BLI-GA-TOI-RE, d'autant que les problèmes d'ouverture de fichier sont bien plus courants que les problèmes d'allocation de mémoire dans le tas !

ligne 8 Gestion de l'erreur via l'affichage d'un message explicite sur la sortie standard (`printf`).

ligne 14 Utilisation de `fscanf` pour lire un entier. Cela suppose que le fichier commence par une chaîne de caractères pouvant être lue comme une valeur entière. La valeur `EOF` (*End Of File*) est renvoyée si la fin du fichier est atteinte avant que la lecture attendue ait pu être réalisée. En cas de succès, la fonction renvoie le nombre de lectures correctement réalisées.

ligne 16 Fermeture du flux associé au pointeur de fichier `fd`.

Fichiers fournis

Les deux versions de dictionnaire fournies sont issues des dictionnaires utilisés par le projet libre Grammalecte.
Les fichiers contiennent tous les deux une première ligne qui indique le nombre de mots, puis une ligne contenant uniquement un tiret, puis un mot par ligne. Sur chaque ligne de mot, le mot est écrit au début de la ligne puis il est séparé d'information sur la nature du mot par le signe `/`.