# Programmation : TP 1

**Objectifs** : [O'Caml, C, UNIX] Utilisation de l'environnement. Édition, compilation, exécution d'un programme.

## Les documentations de référence

Les documentations en ligne sont consultables via votre navigateur Netscape, à l'URL suivante :

`http://rafale/`

Vous pouvez également consulter l'aide en ligne avec la commande UNIX `man`, pour chacune des instructions C, à condition de connaître son nom. Par exemple pour obtenir de l'aide sur l'instruction `printf`, tapez `man printf` dans une fenêtre shell.

Pour le langage C, le livre de référence est : *Le langage C, Norme ANSI, 2ème édition, Brian W. Kernighan, Denis M. Ritchie, DUNOD.*

A signaler également, pour le langage O'Caml, le livre de Pierre Weis et Xavier Leroy : *Le langage Caml*, chez *InterEditions*, très instructif même si la syntaxe peut avoir légérement évolué depuis l'édition de ce livre.

## Quelques mots sur la compilation

Tous vos programmes, que ce soit en langage C ou O'Caml ou tout autre langage, sont au départ écrits sous forme de texte et sauvegardés dans des *fichiers de code source* (extensions `.c` ou `.ml`), créés ou modifiés avec un éditeur de texte (XEmacs dans ce TP et les suivants).

La compilation est une succession d'étapes visant à transformer ce texte en un programme, c'est-à-dire un *fichier de code exécutable* par votre machine (une suite d'instructions de code binaire pouvant être chargée dans la pile d'instructions du processeur).

Nous utiliserons les compilateurs `gcc` pour le langage C, et `ocamlc` pour O'Caml.

## Compilation et exécution de programmes C ou O'Caml

1. Connectez vous, ouvrez une fenêtre shell, et créez un répertoire (`C` par exemple) qui vous servira à ranger vos programmes en langage C.
2. Copiez dans ce répertoire le fichier `bonjour.c` situé dans le répertoire `/usr/local/serveur/PROG/C/`.
3. Placez-vous dans votre répertoire C et compilez `bonjour.c`.

   ```
   gcc -o bonjour bonjour.c
   ```

   Comme vous pouvez le constater en visualisant le contenu détaillé de votre répertoire (commande `ls -l`), cette commande produit un *fichier exécutable* appelé `bonjour`.

   **Note :** *Si vous oubliez l'option* `-o nom_executable`, *le compilateur produit un exécutable nommé* `a.out`.

4. Exécutez le programme :

   ```
   ./bonjour
   ```

5. De façon analogue, créez un répertoire `Ocaml` (au même niveau que `C`), récupérez le fichier `bonjour.ml` situé dans `/usr/local/serveur/PROG/Ocaml/`, et compilez-le avec la commande :

   ```
   ocamlc -o bonjour bonjour.ml
   ```

   puis exécutez-le.

   En visualisant le contenu détaillé de votre répertoire (toujours `ls -l`), vous constaterez que la compilation a produit l'exécutable `bonjour`, mais également des *fichiers de code objet* (`bonjour.cmo` et `bonjour.cmi`), qui sont le résultat d'une étape intermédiaire de la compilation (ceci sera détaillé en TP 9).

## XEmacs : premiers pas

Pour commencer, nous allons simplement visualiser avec un éditeur de texte (XEmacs) les fichiers `bonjour.c` et `bonjour.ml`.

1. Placez-vous dans votre répertoire racine (commande `cd ~`), puis lancez l'éditeur en tâche de fond :

   ```
   xemacs &
   ```

2. Chargez le fichier `bonjour.c` en tapant `^x ^f` puis en entrant le chemin du fichier. Le signe `^` représente la touche CTRL.
3. Afin de pouvoir visualiser le deuxième fichier, coupez en deux votre fenêtre XEmacs avec le racourci clavier `^x 2` (attention, pas `^x ^2`).
4. Chargez le fichier `bonjour.ml`, et examinez le code des deux programmes.

   Remarquons qu'en C, l'argument de la fonction d'affichage est placé entre parenthèses juste après le nom de la fonction, alors qu'en O'Caml l'argument est simplement séparé du nom de la fonction par un espace.

   Les instructions `#include` en C, et `open` en O'Caml permettent de faire appel à des fonctions définies dans des librairies (ici les fonctions `printf`).
5. Lorsque vous avez fini d'examiner ces deux programmes, supprimez les *buffers* XEmacs les affichant (`^x k` dans chacune des deux fenêtres), puis revenez à un mode d'affichage sur une seule fenêtre (`^x 1`).

## Modification de programmes

1. Placez-vous dans votre répertoire C et récupérez le fichier `iznogoud.c` dans le répertoire `/usr/local/serveur/PROG/C/`.
2. Essayez de le compiler et de l'exécuter.
3. Ouvrir le fichier `iznogoud.c` sous XEmacs. Qu'est censé faire ce programme ?
4. Configurer votre XEmacs pour afficher les numéros de ligne et de colonne. Pour cela, ouvrir, toujours avec XEmacs, le fichier `.emacs` dans votre répertoire racine et rajouter les deux lignes suivantes :

   ```
   (line-number-mode t)
   (column-number-mode t)
   ```

   Sauvegarder le fichier (raccourci clavier `^x ^s`), fermer XEmacs (`^x ^c`) et relancer le.
5. Corriger les erreurs dans `iznogoud.c`, sauvegarder vos modifications, et recompiler jusqu'à obtenir un programme correct :

   ```
   gcc -o iznogoud -W -Wall iznogoud.c
   ```

   Les options `-W` et `-Wall` permettent l'affichage d'avertissements (*warnings*) lorsque la syntaxe utilisée n'est pas "propre". Idéalement, il ne doit rester que les *warnings* suivants :

   ```
   unused parameter ‘argc’
   unused parameter ‘argv’
   ```

   `argc` est le nombre d'arguments à passer à l'exécutable `iznogoud`, et `argv` le tableau d'arguments. Vous pouvez éventuellement les enlever puisqu'ils ne sont pas utilisés.

   **Note :** *pour avoir le détail des options du compilateur, vous pouvez consulter le manuel en ligne en tapant* `man gcc` *dans une fenêtre shell.*
6. De la même façon, récupérer le programme Ocaml `ipabon.ml` dans le répertoire `/usr/local/serveur/PROG/Oca` et le corriger.
7. Lorsque vous avez fini, quittez XEmacs et terminez votre session.