Programmation : TP 8

Objectifs : [O'Caml] interface, modularité ; [C] *Header*, cpp, compilation séparée, *Makefile*.

Introduction

Pour des programmes complexes, on sépare le code source en plusieurs fichiers, par groupes de fonctionnalités, par concepts... Cette *modularité* (regroupement du code en *modules*) facilite le développement structuré, la robustesse des programmes, le travail en équipe ou encore la maintenance des programmes.

[O'Caml]

1. Copier dans un fichier `sigma.ml` les fonctions $\sum$ du TP 6. Compiler le fichier, **sans** faire l'édition de liens (option `-c`).
2. Dans un fichier `fact.ml`, écrire la fonction factorielle du TP 6, en faisant appel au module `Sigma`. Compiler.
3. Effectuer l'édition de liens avec les deux *objets* `sigma.cmo` et `fact.cmo`. Exécuter.
4. Créer une interface pour le module `Sigma`, `sigma.mli` ne contenant que la déclaration de la fonction utilisée dans `Fact`. Une déclaration en O'Caml a la forme suivante (fonction à deux arguments renvoyant la somme de deux entiers) :

   ```
   val sum : int -> int -> int;;
   ```

   Une bonne façon d'obtenir le type d'une fonction est de la compiler dans le *toplevel*. Par exemple :

```
#let f = fun x y g -> (y, g x);;
val f : 'a -> 'b -> ('a -> 'c) -> 'b * 'c = <fun>
```

5. Compiler l'interface avec

   ```
   ocamlc sigma.mli
   ```

   qui produit `sigma.cmi` puis les fichiers `.ml` (avec l'option `-c`). Terminer par l'édition de liens. Exécuter.
6. Écrire un `Makefile` pour compiler l'ensemble des fichiers. Y ajouter une cible `clean` :

   ```
   clean :
   <tabulation>rm -f *.cm*
   ```

   Utiliser avec la commande `make clean`.

[C]

On choisit de représenter un polynôme à coefficients entiers par un tableau contenant son degré et ses coefficients. Ainsi, le polynôme X^3+5X+8 sera représenté par le tableau `{3, 8, 5, 0, 1}` : l'élément d'indice 0 correspond au degré du polynôme et l'élément d'indice i est le coefficient de X^{i-1}.

1. Dans un fichier `poly.h` définir deux *macros* :
   (a) `Degre(p)` renvoie le degré d'un polynôme `p` :
   (b) `Coeff(p, i)` renvoie le coefficient de $X^{\texttt{i}}$ de `p`.

   Pour utiliser ces macros, ce fichier sera inclus dans les autres fichiers écrits par la suite.
2. Dans un fichier `calc.c`, définir la fonction produit d'un polynôme par un scalaire de profil suivant :

   ```
   int *scalp(int, int*);
   ```

On utilisera une fonction intermédiaire d'allocation **qui ne sera pas publique** (utiliser le qualificatif `static`). On utilisera donc un polynôme intermédiaire que l'on retournera plutôt que de modifier *en place* celui passé en argument. Écrire l'en-tête correspondant : `calc.h`.

3. Dans un fichier `print.c`, définir une fonction d'affichage d'un polynôme. Écrire l'en-tête correspondant `print.h`.
4. Dans un troisième fichier `tp8.c`, écrire une fonction `main()` qui fait appel aux fonctions définies précédemment.
5. Écrire un `Makefile` et compiler le tout avec un simple `make`.