

CaML - Licence Pro - 2009-10

TP1

28 février 2010

Question 1 *Installation de OCaml :*

1. aller à la page <http://caml.inria.fr/caml-light/release.fr.html>
2. télécharger le binaire pour Windows
3. ???
4. ???

Lancer l'interprète à l'aide de la commande `ocaml`.

Question 2 (*Syntaxe élémentaire*) Calculer $1+1$ en Caml :

`1+1 ; ;`

Remarquer le double point-virgule qui ordonne à Caml d'exécuter l'opération demandée. Essayer sans le double point-virgule...

On peut mettre des espaces autant que l'on veut, et même des parenthèses :

`(((1)) + 1) ; ;`

Question 3 (*Types de base*) Caml connaît un certain nombre de types de constantes. Essayer successivement :

- `2 ; ;`
- `2.5 ; ;`
- `2. ; ;`
- `-2 ; ;`
- `true ; ;`
- `'a' ; ;`
- `"coucou" ; ;`

Question 4 (*Calcul arithmétique*) Essayer les lignes suivantes :

- `2+5 ; ;`
- `2*.5 ; ;`
- `-2/.0.5 ; ;`
- `-2 ** 5 ; ;`
- `-2. ** 5. ; ;`

Question 5 (Booléens) Essayer les lignes suivantes :

- `1 < 2 ; ;`
- `(4.6 + .0.2) < 4.7 ; ;`
- `'a' < 'b' ; ;`
- `"ga" = "bu" ; ;`
- `"aab" < "aac" ; ;`

Question 6 (Conditionnelles) Taper les lignes suivantes :

- `if 3*5 < 16 then "yes" else "no" ; ;`
- `let test = ("caml" > "autre langage") ; ;`
- `if test then 1+1 else 0 ; ;`
- `if true then 't' else "false" ; ;`

Fermez l'environnement à l'aide de la commande `#quit ; ;`.

Créez un fichier "tp1.ml". Dans la suite du TP, nous n'utiliserons l'interprète que pour lire ce fichier, à l'aide de l'instruction `open`.

Nous allons à présent commencer à créer un petit programme de manipulation de liste :

Question 7 Ecrire une fonction `max` qui retourne le maximum de deux éléments, dans `tp1.ml`.

Ecrire la commande `ocaml tp1.ml`.

Ecrire la commande `ocaml < tp1.ml`. Quel est le type de cette fonction ? Commenter.

Dans `tp1.ml`, écrire un appel à `max`. Relancer les commandes précédentes.

Question 8 Dans votre fichier, rajouter les fonctions suivantes :

1. `double` qui calcule le double d'un élément en entrée.
2. `plus` qui calcule la somme de deux entiers.
3. `min10` qui retourne le minimum entre 10 et un autre élément (de quel type ?).
4. `max10` qui retourne le maximum entre 10 et un autre élément (de quel type ?).

On rappelle qu'une liste est un type prédéfini en OCaml et que tous ces éléments doivent être de même type.

Question 9 Ecrire une fonction `isEmpty` qui retourne un booléen selon que la liste est vide ou non. On donnera trois versions de cette fonction :

1. une utilisant la commande `if ... then ... else ...`
2. une autre utilisant le `pattern matching`
3. ...la plus concise possible

Question 10 Ecrire une fonction `length` qui calcule la taille d'une liste.

Question 11 *Ecrire une fonction `list_max` qui retourne l'élément maximum d'une liste. On cherchera le code le plus concis possible, sans introduire de variable locale à la fonction.*

Remarque : Afin d'éviter des problèmes sur le typage, on pourra rajouter le cas : `[] -> failwith "Liste vide !"`.

Question 12 (Avantage du fonctionnel) *On désire généraliser cette opération à d'autres opérations telles que le calcul de la somme des éléments d'une liste. Pour ceci, on se donne une fonction `map` de type `'a list -> ('a -> 'a -> 'a) -> 'a` comme suit :*

```
let rec map l f = match l with
| [a] -> a
| a : :k -> f a (map k f)
| [] -> failwith "Liste vide !";;
```

1. Commenter cette fonction : quels sont ces arguments ? Quel est le type de retour ?
2. Ajouter cette fonction à votre fichier "tp1.ml".
3. Réécrire la fonction `list_max` à l'aide de cette fonction.
4. Ecrire la fonction `somme_liste` qui calcule la somme des éléments d'une liste à partir de cette fonction.

Question 13 (Avantage du fonctionnel (bis)) *On désire à présent écrire des fonctions qui, à partir d'une liste `l1`, retourne une nouvelle liste `l2` contenant tous les éléments de la liste `l1` modifiés par une fonction `f`.*

1. Pour ceci, on veut écrire une fonction `map` de type `'a list -> ('a -> 'a -> 'a) -> 'a`.
2. Ecrire la fonction `double_liste` qui prend une liste `l` en entrée, et renvoie la liste dont tous les éléments sont le double de l'élément correspondant dans la liste d'origine.
3. Ecrire la fonction `somme_liste` qui calcule la somme des éléments d'une liste à partir de cette fonction.
4. De même avec les fonctions `min10` et `max10`.

Question 14 (Tri par insertion) *On écrit ce tri en deux étapes :*

1. écrire une fonction `insert` qui insère un élément à sa place dans une liste triée par ordre décroissant.
2. écrire la fonction `tri_insert` qui réalise le tri par insertion.