Input/Output en OCaml

Afin de lire/d'écrire dans des fichiers en OCaml, on utilise les deux fonctions :

- `output_string : out_channel -> string -> unit` qui se comporte comme `print_string` sauf qu'elle prend comme argument supplémentaire un descripteur du fichier dans lequel elle doit écrire (de type `out_channel`).
- `input_line : in_channel -> string` prenant en argument un descripteur du fichier dans lequel elle doit lire (de type `in_channel`) et retournant la prochaine ligne du fichier.

Les descripteurs de fichiers (de type `out_channel` pour l'écriture et `in_channel` pour la lecture) sont obtenus par ouverture de fichiers :

- en lecture, au moyen d'un appel à la fonction `open_in : string -> in_channel` ;
- en écriture, au moyen d'un appel à la fonction `open_out : string -> out_channel`.

Le typage assure qu'un fichier ouvert en lecture ne peut qu'être lu, tandis qu'un fichier ouvert en écriture ne peut être qu'écrit.
Après usage, on ferme un fichier ouvert en lecture avec la fonction `close_in : in_channel -> unit` et un fichier ouvert en écriture avec la fonction `close_out : out_channel -> unit`.

Par exemple le programme OCaml ci-dessous, lorsqu'il est exécuté en présence d'un fichier `a.txt` de la forme :

```
toto
3 4
```

produit un fichier `b.txt` de la forme :

```
(toto)4 3
```

```
let () =
  let fp1 = open_in "a.txt" in
  let fp2 = open_out "b.txt" in
  let buffer = input_line fp1 in
  output_string fp2 ("(" ^ buffer ^ ")");
  let ligne2 = input_line fp1 in
  let tmp1 = int_of_string (String.sub ligne2 0 1) in
  let tmp2 = int_of_string (String.sub ligne2 2 1) in
  output_string fp2 (string_of_int tmp1);
  output_string fp2 " ";
  output_string fp2 (string_of_int tmp2)
```