LES ERREURS

On peut matcher les erreurs :

```
match maFct | with
  | Error -> Failwith ("msg perso")
  | Ok    -> ...
```

Quelques types d'exception :

- Not_Found
- Failure "msg"
- Division_by_zero
- Stack_overflow
- Out_of_memory
- Sys.break
- ⋮

} Système

On peut en ajouter :

```
exception Echec
```

crée une exception nommée Echec

```
exception E of int
```

↳ que des types concrets. (pas de 'a)

crée une exception avec un entier en argument

Le type exn est extensible et il contient toutes les erreurs y-compris celles qu'on ajoute

(Not_found, E 10)
→ exn * exn = (Not_found, E 10)

pour lever une exception (throw en java)

if x < 0 then raise (E 10) else x + 42.

let Failwith msg = raise (Failure msg)
est l'implémentation de Failwith en Ocaml.

let F x y =
try x / y with Division_by_zero → - 1

test si y = 0, si on renvoie x/y

try ...
with
| Mon_exception → ...
| Not_found → ...

On peut gérer plusieurs exception

Comme en Java, on crée une fonction auxiliaire qui calcule et raise éventuellement.
Puis une fonction qui l'appelle, dans un try ... with Mon_exception → ...