

Exceptions

Les exceptions de ML ont servi de modèle pour celles du langage C++.

```
exception Perdu
```

```
let rec cherche_la_clé k = function
  (h,v)::t -> if h = k then v else cherche_la_clé k t
| [] -> raise Perdu

let k =
  try
    cherche_la_clé "Louis" [("Georges",14); ("Louis",5)]
  with Perdu -> 10
```

Exercice 5 Réécrire le programme sans user d'exceptions.

Exceptions (suite)

Syntaxe

| | |
|------------------------|--|
| Définition (phrase) | <code>exception C [of t];;</code> |
| Lancement (expression) | <code>raise e;;</code> |
| Filtrage (expression) | <code>try e with $p_1 \rightarrow e_1 \mid \dots \mid p_n \rightarrow e_n$;;</code> |

Remarquez l'analogie avec le filtrage des valeurs.

Typage

Les exceptions sont toutes de type `exn`, qui peut être considéré comme un type somme ouvert (de nouveaux constructeurs peuvent être ajoutés avec des déclarations `exception`).

Exceptions prédéfinies

The Core Library

| Constructeur | Usage |
|---|-------------------------------------|
| <code>Invalid_argument of string</code> | Argument hors bornes |
| <code>Failure of string</code> | Fonction indéfinie pour un argument |
| <code>Not_found</code> | Échec de fonctions de recherche |
| <code>Match_failure of ...</code> | Échec de filtrage |
| <code>End_of_file</code> | Fin de fichier |

Sémantique des exceptions

- Le type `exn` est le seul type somme *extensible*.
- Le lancement d'une exception arrête l'évaluation et retourne une valeur exceptionnelle (c-à-d. de type `exn`).
- Une exception ne peut être éventuellement filtrée que si l'expression a été encadrée par un bloc `try e with m` :
 - Si l'évaluation de `e` retourne une valeur normale, celle-ci est retournée sans passer par le filtre `m`.
 - Sinon, l'exception est passée au filtre `m`. Si un des motifs p_i filtre l'exception, alors `ei` est évaluée, sinon l'exception est propagée (**Les filtres d'exceptions ne sont pas forcément complets.**).
- On peut observer une exception (c-à-d. la filtrer puis la relancer) :
`try f x with Failure s as x -> prerr_string s; raise x`