Exceptions

ML provides an elegant
exception handling mechanism

1. built-in exceptions
2. partial functions
3. user-defined exceptions
4. exception handling

Built-in Exceptions

```
- 5 div 0;
uncaught exception Div [divide by zero]
  raised at: ...

- chr(500);
uncaught exception Chr
  raised at: ...

- hd(nil: int list);
uncaught exception Empty
  raised at: ...
```

Exceptions for Non-Total Functions

Some functions are naturally undefined for some input. Dealing with that can be awkward:

```
fun lookup_table x nil = NONE
  | lookup_table x ((x',v')::t)
      = if x = x' then SOME v'
                  else lookup_table x t;

(*  ''a -> (''a * 'b) list -> 'b option *)
```

Instead, one may explicitly raise an exception:

```
exception Empty_table;

fun lookup_table x nil = raise Empty_table
  | lookup_table x ((x',v')::t)
      = if x = x' then v'
                  else lookup_table x t;

(*  ''a -> (''a * 'b) list -> 'b  *)
```

Exceptions with Parameters

```
exception Empty_table of string;
fun lookup_table x nil
        = raise Empty_table(x)
  | lookup_table x ((x',v')::t)
        = if x = x' then v'
                    else lookup_table x t;
(* string -> (string * 'a) list -> 'a *)
```

Note: polymorphism of function has been lost

```
uncaught exception Empty_table
  raised at: lookup_table-3.sml:3.32-3.46
/usr/local/bin/sml: Fatal error -- Uncaught
  exception Empty_table with "mary" raised
  at lookup_table-3.sml:3.32-3.46
```

```
- Empty_table;
val it = fn : string -> exn
```

```
- raise Empty_table;
Error: argument of raise is not an exception
```

Exception Handling

Wrap a handler around an exception returning expression:

```
fun lookup_table' x v
      = lookup_table x v
          handle Empty_table(s)
            => (print("Entry␣not␣found:␣");
                print(x);
                print("\n");
                0);
(* string -> (string * int) list -> int *)
```

```
- lookup_table' "ed" [("joe",12),("ed",7)];
val it = 7 : int
```

```
- lookup_table' "mary" [("joe",12),("ed",7)];
Entry not found: mary
val it = 0 : int
```