CAML

$$\#\ \texttt{let}\ \text{sum}\ x\ y = \underbrace{x + y}_{\text{exp}}\ ;;$$

nome 1° arg 2° arg.

$$\text{sum} : \underbrace{'a}_{\text{tipo } x} \to \underbrace{'b}_{\text{tipo } y} \to \underbrace{\cancel{'c}}_{\text{ris.}}$$

$$\text{sum} : \text{int} \to \text{int} \to \text{int}$$

INFERENZA DI TIPO

FUNZIONI DI ORDINE SUPERIORE

Sono funzioni che ricevono come argomento altre funzioni e/o restituiscono come risultato una funzione

```
# sum 3 ;;
- : int -> int = <fun>
# let inc3 = sum 3 ;;
inc3 : int -> int = <fun>
# inc3 8 ;;
- : int = 11
# inc3 25 ;;
- : int = 28
```

```
let sum x y = x+y;;
sum : int -> int -> int
```

```
# let inc1 = sum 1;;
inc1 : int -> int = <fun>
```

```
# inc1 10;;
- : int = 11
# inc1 25;;
- : int = 26
```

```
# let apply x y = x y;;
```

$$apply : \cancel{'a} \to \cancel{'b} \to \cancel{'c}$$

$$(\underbrace{'d}_{int} \to \underset{int}{\overset{}{'e^{*}}}) \to \underbrace{'d} \to {'e}^{*}$$

```
# let g = apply inc1;;
g : int -> int = <fun>
```

```
# let gt x y = x > y ;;            greater than
  gt : int -> int -> bool = <fun>

# let neg = gt 0 ;;
  neg : int -> bool = <fun>

# let h = apply neg ;;
  h : int -> bool = <fun>
```

```
apply : ('a -> 'b) -> 'a -> 'b
neg   :   int -> bool
```

INFERENZA DI TIPO

let $f\ x_1\ x_2 \ldots x_n = Exp$;;

1. STRUTTURA DEL TIPO DI f

$$f : 'a1 \to 'a2 \to \cdots \to 'an \to 'r$$

2. Si ragiona sull'uso di $x1, \ldots, xn$ in Exp per ISTANZIARE (CONCRETIZZARE) 'a1 … 'an (se possibile)

In questo ci si può accorgere che alcune variabili di tipo DEVONO COINCIDERE

FUNZIONE UNCURRIED

Caso particolare in cui abbiamo 1 solo argomento

<u>let</u> $f\ (y_1, \dots, y_n) = Exp$

$$f : \text{'}a \rightarrow \text{'}b$$

$$\Downarrow$$

$$f : \text{'}a1 * \text{'}a2 * \dots * \text{'}an \rightarrow b$$

CONDIZIONALE : espressioni

```
# let abs x = if x>0 then x
                      else -x;;
abs : int -> int = <fun>
```

$$\text{if } C \text{ then } E1 \text{ else } E2$$

exp di tipo bool

devono avere lo stesso tipo

if x>0 then true else 1

errore di tipo

RICORSIONE

\# let rec f ..... = ... f ......

f unbound

let rec pot (x, y) = if y = 0 then 1
else x * pot (x, y-1);;

pot : (int * int) → int = <fun>

$$\text{pot}(x,y) = \begin{cases} 1 & \text{se } y = 0 \\ x * \text{pot}(x, y-1) & \text{altrimenti} \end{cases}$$

$pot(3,2)$

$= \{\text{def. di pot}\}$

$3 * pot(3,\underline{1})$

$= \{\text{def di pot}\}$

$3 * 3 * pot(3,\underline{0})$

$= \{\text{def di pot}\}$

$3 * 3 * 1$

$=$

9

$pot(3,-1)$

$= \{\text{def di pot}\}$

$3 * pot(3,-2)$

$=$

$3 * 3 * pot(3,-3)$

$\vdots$

$pot(x,y) = \text{if } y=0$

$\text{then } 1$

else

$x * pot(x,y-1)$

$$pot : \mathbb{Z} * \mathbb{Z} \longrightarrow \mathbb{Z}$$

Relazione indotta $\prec_{pot}$

$m \in \mathbb{Z}^+$

$$\begin{array}{c} (n, m) \\ | \\ (n, m-1) \\ | \\ \vdots \\ (n, 1) \\ | \\ (n, 0) \end{array}$$

$m \in \mathbb{Z}^-$

$$\begin{array}{c} (n, m) \\ | \\ (n, m-1) \\ | \\ (n, m-2) \\ | \\ \vdots \end{array}$$

$\forall (x,y) \in (\cancel{\mathbb{Z}^+ * \mathbb{Z}^+}\ (\mathbb{N} * \mathbb{N}).\ pot(x,y) = x^y$

- pot è TOTALE su $\mathbb{Z} * \mathbb{N}$
- pot è PARZIALE (non TOTALE) su $\mathbb{Z} * \mathbb{Z}$

Caso base

$pot(x, \emptyset)$

$= \{\text{def di pot, ramo then}\}$

1

$=$

$x^{\emptyset}$

Caso induttivo (x,y) con $y > \emptyset$

$pot(x,y)$

$= \{\text{def. di pot, ramo else}\}$

$x * \underline{pot(x, y-1)}$

$= \{\text{Ip. ind. } (x, y-1) <_{pot} (x,y)\}$

$x * x^{y-1}$

$= x^{y \cancel{-1} \cancel{+1}}$ c.v.d.

DIMOSTRAZIONI PER INDUZIONE BEN FONDATA

let rec pot x y = if y=0 then 1 else x * pot x y-1;;

$\Downarrow$

dimostrazione per ind. b.f. la facciamo

come nella versione uncurryed.

PATTERN MATCHING

$(3, 2)$ $\qquad\qquad$ $(3, x)$

oggetto 1 $\qquad\qquad$ oggetto 2

fare pattern matching su questi due oggetti significa domandarsi: possono COINCIDERE?

$(3, 2)$ $\qquad\qquad$ $(3, x)$

purché $x = 2$

$$(3, 2) \not\longleftrightarrow (4, x)$$

```
let rec pot (x,y) = if y=0 then 1 else x *
                                       pot (x,y-1) ;;
```

$$pot(x,y) = \begin{cases} 1 & 1^\circ \text{ caso } \quad y=0 \\ x * pot(x,y-1) & 2^\circ \text{ caso} \end{cases}$$

```
let rec pot (x,y) = match y with
   0 -> 1 |
   _ -> x * pot (x,y-1) ;;
```

PATTERN MATCHING

$$\underline{\text{match}}\ E\ \underline{\text{with}}$$

valori concreti o variabili

$p_1 \rightarrow E1$ |

$p_2 \rightarrow E2$ |

$\vdots$

$p_n \rightarrow E_n$

tipo uguale

---

if $(E = p_1)$ then $E1$ else

if $(E = p_2)$ then $E2$ else ....

.... if $(E = p_n)$ then E_n else ?

match y+1 with

~~x*3~~ ⟶ _ _ _

numero (costante intera)

variabile (nome, sta per il pattern "generico")

_ (underscore ")

(in tutti gli altri casi)

```
let f x = match x with
    0 -> 1 |
    1 -> 25 |
    _ -> 45 ;;
```

$$f(n) = \begin{cases} 1 & \text{se } x = 0 \\ 25 & \text{se } x = 1 \\ 45 & \text{ALTRIMENTI} \end{cases}$$

```
let f x = match x with
    0 -> 1 |
    1 -> 10 ;;

f : int -> int = <fun>
```

Warning: il p.m. non è ESAUSTIVO

non copre tutti i possibili valori

```
let f x = match x with
    0 -> 1 |
    1 -> 10 |
    _ -> 45 ;;

f : int -> int = <fun>
```

$$g(n) = \begin{cases} 1 & \text{se} \quad n = \boxed{0} \\ 10 & \text{se} \quad \boxed{n > 0} \\ 25 & \text{se} \quad \boxed{n < \emptyset} \end{cases}$$

PATTERN MATCHING

```
match n with
  ∅            ->  1  |
  x when x>0   ->  10
  x when x<0   ->  25
```

```
let g n =   match n with
      0  ->  1    |
 x when x>0  ->  10   |
 x when x<0  ->  25   ;;
```

g : int -> int = <fun>

g (4) pattern matching 4 <~~> 0 NO

4 <~~> x x=4 SI

= 10

g(-3)

-3 <~~> 0 NO

-3 <~~> x x=-3 ma -3 $\not>$ 0 NO

-3 <~~> x x=-3 -3<0 SI

= 25

Sia un nome x che _ sono pattern "generici" che "matchano" qualunque valore

Quando si usa _ e quando si usa x ??

$$\text{match } (x, y) \text{ with}$$

$$(\_, 3) \text{ when } \cancel{\_ > 0} \longrightarrow y + 5$$

$$(z, 3) \text{ when } z > 0 \longrightarrow y + 5$$

match E with

$p1 \to E1$

$p2 \to E2$

$\vdots$

$pn \to En$

$\_ \to E_{n+1}$

altrimenti

match n with

$0 \to n+1$ 1 |

$\to x$ when $x > 0 \to \cancel{x}+2$ |

$n+2$

$\to \boxed{x \to x+3\ ;;}$

da qui in poi (fino alla |) n e x sono SINONIMI

? cosa è x ??

x unbound

$\to$ _ $\to n+3\ ;;$

PATTERN MATCHING : COPPIE

- `let f (x, y) = x + y ;;`

- `let f z = match z with`
  `(x, y) -> x + y ;;`

LISTE : sequenze di oggetti

```
let pot (x, y) = match y with
     0 -> 1 |
n  when n>0 -> x * pot (x, y-1) |
     _ -> -1 ;;

pot : int * int -> int = <fun>
```

$$f(x,y) = \begin{cases} x^y & \text{se } y \geq 0 \\ -1 & \text{altrimenti} \end{cases}$$