

QUIZ #1 ↓

Swap 3 5
Swap 3.0 5

(5, 3)
(5, 3.0)

let swapInt (x: Int) (y: Int) = (y, x)
let swapReal (x: Float) (y: Float) = (y, x)

let swapIntReal (x: Float) (y: Int) = (y, x)
(x: Int) (y: Float)

Type polymorphism

let swap (x: 'a) (y: 'b) : 'b * 'a = (y, x)

val swap : 'a * 'b → 'b * 'a

let append-to-string ((x: 'a), (s: string),
(convert: 'a → string)): string =
(convert x) ^ " " ^ s

'a * string * ('a → string) → string

append-to-string (3110, "class", string-of-int)
"3110 class"

Variant types

type answer = Yes | No | Maybe

Constructors

selectors

type eitherPoint = TwoD of float * float
| ThreeD of float * float * float

TwoD (3.1, 4.6)

let lastTwo (p: eitherPoint): float * float =
match p with
TwoD (x, y) → (x, y)
| ThreeD (x, y, z) → (y, z)

int List

Foo Barney

type intList = Nil | Cons of (int * intList)

Nil

Cons (1, Nil)

Cons (2, Cons (1, Nil))

let alist = Cons (1, Nil)

let blist = Cons (3, alist)

let rec length (lst: intList): int =
match lst with

Nil → 0

| Cons (h, t) → length (t) + 1

let rec is-empty (lst: intList): bool =
~~match~~ match lst with

Nil → true

| Cons (_, _) → false

let inc (x: int): int = x + 1

let square (x) = x * x

[a; b; c] [a+1; b+1; c+1]
let rec addone_all (lst: intList): intList =
 match lst with

Nil → Nil

| Cons(h, t) → Cons(inc(h), addone_all(t))

let rec square_all (lst: intList): intList =

match lst with

Nil → Nil

| Cons(h, t) → Cons(square(h), square_all(t))

let rec do_f_to_all ((f: int → int), (lst: intList)): intList =
 match lst with

Nil → Nil

| Cons(h, t) → Cons(f(h), do_f_to_all(f, t))

let square_all (lst: intList): intList =

do_f_to_all(square, lst)

(do_f_to_all((fun x → ~~x~~
z → z * z), lst))

let rec sum (lst: intList): int =
 match lst with

Nil → 0

| Cons(h, t) → h + sum(t)

↑
*