Summary of polymorphism

- Subtype
- Parametric
- Bounded
- F-bounded

Back to OCaml

- Polymorphic types allow us to reuse code
- However, not always obvious from staring at code
- But... Types never entered w/ program!

Type inference

aka: how in the world does Ocaml figure out all the types for me ???

Example 1

```ocaml
let x = 2 + 3;;
let y = string_of_int x;;
```

Example 2

```ocaml
let x = 2 + 3;;
let inc y = x + y;;
```

Example 2

```ocaml
let x = 2 + 3;;
let inc y = x + y;;
```
Example 3

```ml
let foo x =
  let (y, z) = x in
  z - y;;
```

Example 4

Example 4

ML doesn’t know what the function does, or even that it terminates. ML only knows its type!

Example 5

Example 5
Inferring types with ‘a

- Introduce unknown type vars
- Figure out equalities that must hold, and solve these equalities
- Remaining types vars get a forall and thus become the ‘a, ‘b, etc.

Example 6

```ml
let compose (f, g) x = f (g x)
```

Example 7

```ml
let rec fold f cur l =
  match l with
  | [] -> cur
  | h::t -> fold f (f h cur) t
```

Midterm Practice