

Objective Caml version 3.06

```
# let l = ref [1];;
val l : int list ref = {contents = [1]}

# l := 2 :: !l;;
- : unit = ()
```

```
# l;;
- : int list ref = {contents = [2; 1]}
```

```
# let l = ref [];;
val l : 'a list ref = {contents = []}
```

```
# !l;;
- : 'a list = []
```

```
# let l1 = 1 :: !l
and l2 = true :: !l;;
  Characters 34-36:
  and l2 = true :: !l;;
                    ^^
```

This expression has type int list but is here used with type bool list

```
# l := 2 :: !l;;
- : unit = ()
```

```
# !l;;
- : int list = [2]
```

```
# l := [];;
- : unit = ()
```

```
# !l;;
- : int list = []
```

```
# l := true :: !l;;
  Characters 5-9:
  l := true :: !l;;
  ^^^^^
```

This expression has type bool but is here used with type int

```
# l;;
- : int list ref = {contents = []}
```

```
# let x = ref (fun x -> x);;
val x : ('a -> 'a) ref = {contents = <fun>}
```

```
# x := fun x -> x + 1;;
- : unit = ()
```

```
# x;;
- : (int -> int) ref = {contents = <fun>}
```

```
#
```