```
Last login: Fri Feb  2 13:24:42 on ttys003
carbon:$ utop
──────────────────────────────────────────────────────────────────────────────
elcome to utop version 2.0.2 (using OCaml version 4.06.0)
──────────────────────────────────────────────────────────────────────────────

Type #utop_help for help about using utop.

─( 15:44:18 )─< command 0 >──────────────────{ counter: 0 }─
utop # #use "find_and_lookup.ml";;
val m : (string * int) list =
  [("dog", 1); ("chicken", 2); ("dog", 3); ("cat", 5)]
val lookup_all : 'a -> ('a * 'b) list -> 'b list = <fun>
File "find_and_lookup.ml", line 16, characters 23-34:
Error: Unbound value find_all_by
─( 15:44:25 )─< command 1 >──────────────────────────────{ counter: 0 }
─( 15:44:25 )─< command 1 >──────────────────────────────{ counter: 0 }─
utop # #use "find_and_lookup.ml";;
val m : (string * int) list =
  [("dog", 1); ("chicken", 2); ("dog", 3); ("cat", 5)]
val lookup_all : 'a -> ('a * 'b) list -> 'b list = <fun>
val find_all_by : ('a -> 'b -> bool) -> 'a -> 'b list -> 'b list =
  <fun>
val find_all_with'' : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_with' : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_with' : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_with : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_by' : ('a -> 'b -> bool) -> 'a -> 'b list -> 'b list
=
  <fun>
val find_all_with'' : ('a -> bool) -> 'a list -> 'a list = <fun>
─( 15:44:25 )─< command 2 >──────────────────────────────{ counter: 0 }─
utop # find_all_by ;;
- : ('a -> 'b -> bool) -> 'a -> 'b list -> 'b list = <fun>
─( 15:44:54 )─< command 3 >──────────────────────────────{ counter: 0 }─
utop #  let apply f a = f a  ;;
val apply : ('a -> 'b) -> 'a -> 'b = <fun>
─( 15:45:00 )─< command 4 >──────────────────────────────{ counter: 0 }─
utop # compare ;;
- : 'a -> 'a -> int = <fun>
─( 15:47:04 )─< command 5 >──────────────────────────────{ counter: 0 }─
utop # compare 4 5 ;;
- : int = -1
─( 15:53:23 )─< command 6 >──────────────────────────────{ counter: 0 }─
```

```
utop # #use "find_and_lookup.ml";;
val m : (string * int) list =
  [("dog", 1); ("chicken", 2); ("dog", 3); ("cat", 5)]
val lookup_all : 'a -> ('a * 'b) list -> 'b list = <fun>
val find_all_by : ('a -> 'b -> bool) -> 'a -> 'b list -> 'b list =
  <fun>
val find_all_with'' : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_with' : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_with' : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_with : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_by' : ('a -> 'b -> bool) -> 'a -> 'b list -> 'b list
=
  <fun>
val find_all_with'' : ('a -> bool) -> 'a list -> 'a list = <fun>
val take_while : 'a list -> ('a -> bool) -> 'a list = <fun>
-( 15:53:23 )-< command 7 >-------------------------------{ counter: 0 }-
utop # take_while [4;5;6;7;8] (fun x -> x < 6) ;;
- : int list = [4; 5]
-( 15:56:29 )-< command 8 >-------------------------------{ counter: 0 }-
utop # #use "find_and_lookup.ml";;
val m : (string * int) list =
  [("dog", 1); ("chicken", 2); ("dog", 3); ("cat", 5)]
val lookup_all : 'a -> ('a * 'b) list -> 'b list = <fun>
val find_all_by : ('a -> 'b -> bool) -> 'a -> 'b list -> 'b list =
  <fun>
val find_all_with'' : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_with' : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_with' : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_with : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_by' : ('a -> 'b -> bool) -> 'a -> 'b list -> 'b list
=
  <fun>
val find_all_with'' : ('a -> bool) -> 'a list -> 'a list = <fun>
val take_while : 'a list -> ('a -> bool) -> 'a list = <fun>
val drop_while : 'a list -> ('a -> bool) -> 'a list = <fun>
-( 15:56:56 )-< command 9 >-------------------------------{ counter: 0 }-
utop # drop_while [5;6;7;8;6;5;3;2;1] (fun x -> x > 4)
;;
- : int list = [3; 2; 1]
-( 16:03:01 )-< command 10 >------------------------------{ counter: 0 }-
utop # let flip f a b = f b a ;;
val flip : ('a -> 'b -> 'c) -> 'b -> 'a -> 'c = <fun>
-( 16:03:33 )-< command 11 >------------------------------{ counter: 0 }-
utop # let tac = flip (^) ;;
```

```
val tac : string -> string -> string = <fun>
-( 16:07:52 )-< command 12 >-------------------------{ counter: 0 }-
utop # (^) "hello" " world" ;;
- : string = "hello world"
-( 16:08:05 )-< command 13 >-------------------------{ counter: 0 }-
utop # tac "hello" " world" ;;
- : string = " worldhello"
-( 16:08:16 )-< command 14 >-------------------------{ counter: 0 }-
utop # #use "find_and_lookup.ml";;
val m : (string * int) list =
  [("dog", 1); ("chicken", 2); ("dog", 3); ("cat", 5)]
val lookup_all : 'a -> ('a * 'b) list -> 'b list = <fun>
val find_all_by : ('a -> 'b -> bool) -> 'a -> 'b list -> 'b list =
  <fun>
val find_all_with'' : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_with' : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_with' : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_with : ('a -> bool) -> 'a list -> 'a list = <fun>
val find_all_by' : ('a -> 'b -> bool) -> 'a -> 'b list -> 'b list
=
  <fun>
val find_all_with'' : ('a -> bool) -> 'a list -> 'a list = <fun>
val take_while : 'a list -> ('a -> bool) -> 'a list = <fun>
val drop_while : 'a list -> ('a -> bool) -> 'a list = <fun>
val compose : ('a -> 'b -> 'c) -> 'a -> 'b -> 'c = <fun>
-( 16:08:22 )-< command 15 >-------------------------{ counter: 0 }-
utop #
```

Arg	Array	ArrayLabels	Assert_failure	Bigarray	Buffer	Bytes	Bytes