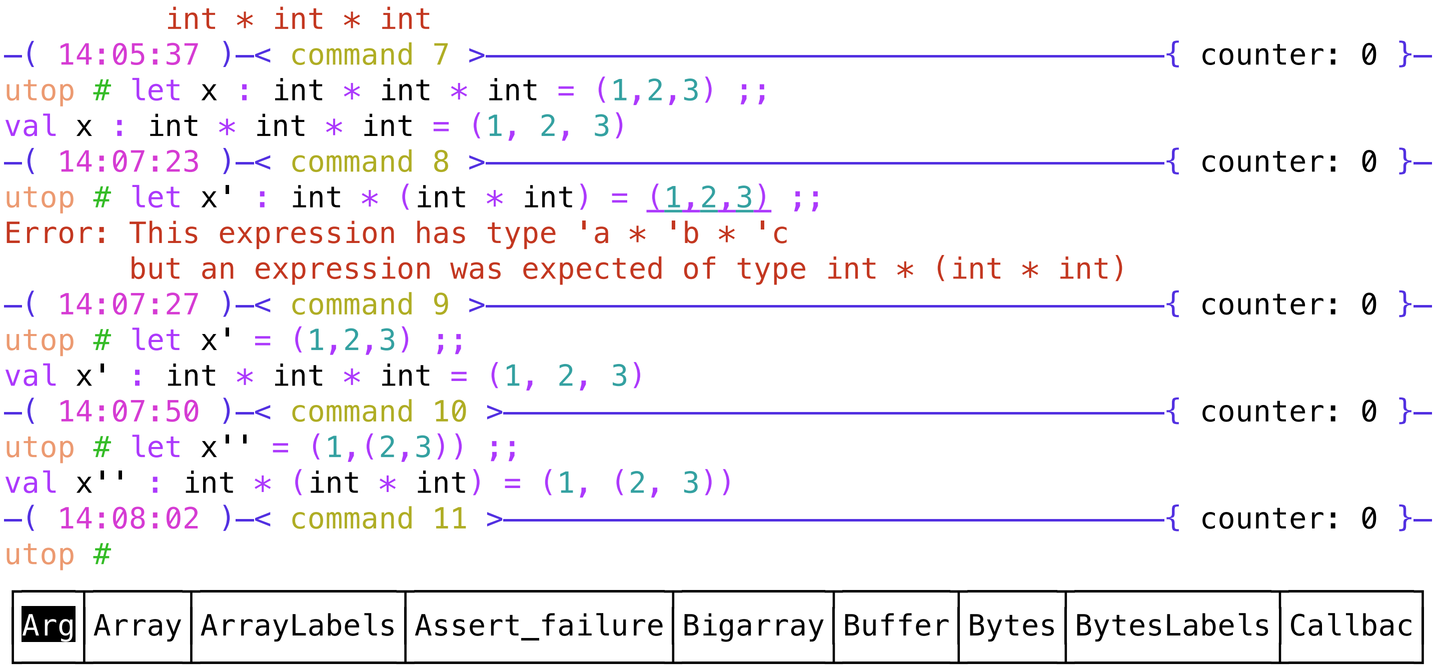
```
Last login: Fri Jan 26 15:39:37 on ttys006
carbon:$ utop
───────────┬──────────────────────────────────────────────────────────────┬───────────
           │ Welcome to utop version 2.0.2 (using OCaml version 4.06.0)! │
           └──────────────────────────────────────────────────────────────┘

Type #utop_help for help about using utop.

─( 13:34:52 )─< command 0 >──────────────────────────────────────{ counter: 0 }─
utop # #use "simple.ml";;
val power : int -> float -> float = <fun>
val cube : float -> float = <fun>
val gcd : int -> int -> int = <fun>
val sum : int list -> int = <fun>
val all : bool list -> bool = <fun>
val even2ways : int list -> bool = <fun>
val is_even : int -> bool = <fun>
File "simple.ml", line 47, characters 6-10:
Warning 26: unused variable even.
val even2ways_better : int list -> bool = <fun>
val string_concat : string -> string list -> string = <fun>
val is_empty : 'a list -> bool = <fun>
val length : 'a list -> int = <fun>
File "simple.ml", line 81, characters 2-32:
Warning 8: this pattern-matching is not exhaustive.
Here is an example of a case that is not matched:
[]
val head : 'a list -> 'a = <fun>
val first : 'a * 'b * 'c -> 'a = <fun>
val first' : 'a * 'b * 'c -> 'a = <fun>
val first'' : 'a * 'b * 'c -> 'a = <fun>
val m : (string * int) list =
  [("dog", 1); ("chicken", 2); ("dog", 3); ("cat", 5)]
val lookup_all : 'a -> ('a * 'b) list -> 'b list = <fun>
val lookup_all' : 'a -> ('a * 'b) list -> 'b list = <fun>
─( 13:34:52 )─< command 1 >──────────────────────────────────────{ counter: 0 }─
utop # lookup_all "dog" m ;;
- : int list = [1; 3]
─( 13:34:55 )─< command 2 >──────────────────────────────────────{ counter: 0 }─
utop # lookup_all' "dog" m ;;
- : int list = [1; 3]
─( 13:35:02 )─< command 3 >──────────────────────────────────────{ counter: 0 }─
utop # 8;;
- : int = 8
─( 13:35:09 )─< command 4 >──────────────────────────────────────{ counter: 0 }─
utop # [ (1,'c') ] ;;
- : (int * char) list = [(1, 'c')]
─( 14:04:39 )─< command 5 >──────────────────────────────────────{ counter: 0 }─
utop # (1, ['c' ] )  ;;
- : int * char list = (1, ['c'])
─( 14:05:21 )─< command 6 >──────────────────────────────────────{ counter: 0 }─
utop # let x : int * int * int = (1,2.3) ;;
Error: This expression has type 'a * 'b but an expression was expected of type
```



```
         int * int * int
─( 14:05:37 )─< command 7 >──────────────────────────────────────{ counter: 0 }─
utop # let x : int * int * int = (1,2,3) ;;
val x : int * int * int = (1, 2, 3)
─( 14:07:23 )─< command 8 >──────────────────────────────────────{ counter: 0 }─
utop # let x' : int * (int * int) = (1,2,3) ;;
Error: This expression has type 'a * 'b * 'c
       but an expression was expected of type int * (int * int)
─( 14:07:27 )─< command 9 >──────────────────────────────────────{ counter: 0 }─
utop # let x' = (1,2,3) ;;
val x' : int * int * int = (1, 2, 3)
─( 14:07:50 )─< command 10 >─────────────────────────────────────{ counter: 0 }─
utop # let x'' = (1,(2,3)) ;;
val x'' : int * (int * int) = (1, (2, 3))
─( 14:08:02 )─< command 11 >─────────────────────────────────────{ counter: 0 }─
utop #
```

│Arg│Array│ArrayLabels│Assert_failure│Bigarray│Buffer│Bytes│BytesLabels│Callbac│