```
Last login: Fri Mar  9 13:24:52 on ttys004
carbon:$ cd Sample\ Programs/Sec_10_3-35pm/
carbon:$ utop
──────────────────────────────────────────────────────────────
e to utop version 2.0.2 (using OCaml version 4.
──────────────────────────────────────────────────────────────

Type #utop_help for help about using utop.

─( 15:54:42 )─< command 0 >──────{ counter: 0 }─
utop # #use "cond.ml";;
val cond : bool -> 'a -> 'a -> 'a = <fun>
val sumton : int -> int = <fun>
─( 15:54:42 )─< command 1 >──────{ counter: 0 }─
utop # sumton 4 ;;
Stack overflow during evaluation (looping recursion?).
─( 15:55:03 )─< command 2 >──────────────────────{ counter: 0 }─
utop # #use "lazy.ml";;
type 'a lazee = 'a hidden ref
and 'a hidden = Value of 'a | Thunk of (unit -> 'a)
val delay : (unit -> 'a) -> 'a lazee = <fun>
val force : 'a lazee -> unit = <fun>
val demand : 'a lazee -> 'a = <fun>
type 'a stream = Cons of 'a * 'a stream lazee
val from : int -> int stream = <fun>
step 1
val nats : int stream = Cons (1, {contents = Thunk <fun>})
val head : 'a stream -> 'a = <fun>
val tail : 'a stream -> 'a stream = <fun>
val take : int -> 'a stream -> 'a list = <fun>
─( 15:55:03 )─< command 3 >──────────────────────{ counter: 0 }─
utop # nats ;;
- : int stream = Cons (1, {contents = Thunk <fun>})
─( 16:23:20 )─< command 4 >──────────────────────{ counter: 0 }─
utop # head nats ;;
- : int = 1
─( 16:23:24 )─< command 5 >──────────────────────{ counter: 0 }─
utop # tail nats ;;
step 2
- : int stream = Cons (2, {contents = Thunk <fun>})
─( 16:23:50 )─< command 6 >──────────────────────{ counter: 0 }─
```

```
utop # tail nats ;;
- : int stream = Cons (2, {contents = Thunk <fun>})
-( 16:23:55 )-< command 7 >-------------------------{ counter: 0 }-
utop # nats  ;;
- : int stream =
Cons (1,
 {contents = Value (Cons (2, {contents = Thunk <fun>}))})
-( 16:24:14 )-< command 8 >-------------------------{ counter: 0 }-
utop # take 5 nats ;;
step 3
step 4
step 5
step 6
- : int list = [1; 2; 3; 4; 5]
-( 16:24:25 )-< command 9 >-------------------------{ counter: 0 }-
utop # nats ;;
- : int stream =
Cons (1,
 {contents =
   Value
    (Cons (2,
      {contents =
        Value
         (Cons (3,
           {contents =
             Value
              (Cons (4,
                {contents =
                  Value
                   (Cons (5,
                     {contents =
                       Value
                        (Cons (6, {contents = Thunk <fun>}))}
))})))})))})))})
-( 16:25:04 )-< command 10 >------------------------{ counter: 0 }-
utop # #use "cond.ml";;
val cond : bool -> 'a -> 'a -> 'a = <fun>
val sumton : int -> int = <fun>
Stack overflow during evaluation (looping recursion?).
-( 16:25:04 )-< command 11 >------------------------{ counter: 0 }-
utop #
```

Arg	Array	ArrayLabels	Assert_failure	Bigarray	Buffer	Bytes	