```
Last login: Wed Feb  7 13:24:54 on ttys003
carbon:$ utop
```

```
Welcome to utop version 2.0.2 (using OCaml version 4.06.0)!

Type #utop_help for help about using utop.

─( 15:36:24 )─< command 0 >──────────────────────────────{ counter: 0 }─
utop # #use "higher_order.ml";;
File "higher_order.ml", line 47, characters 11-13:
Error: Syntax error
─( 15:36:24 )─< command 1 >──────────────────────────────{ counter: 0 }─
utop # #use "higher_order.ml";;
val implode : char list -> string = <fun>
val explode : string -> char list = <fun>
val map : ('a -> 'b) -> 'a list -> 'b list = <fun>
val inc_all_by : int -> int list -> int list = <fun>
val filter : ('a -> bool) -> 'a list -> 'a list = <fun>
val removeABCD : char list -> char list = <fun>
val foldr : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b = <fun>
val f : int -> 'a -> int = <fun>
File "higher_order.ml", line 48, characters 17-22:
Error: Unbound value foldl
Hint: Did you mean foldr?
─( 15:41:21 )─< command 2 >──────────────────────────────{ counter: 0 }─
utop # #use "higher_order.ml";;
val implode : char list -> string = <fun>
val explode : string -> char list = <fun>
val map : ('a -> 'b) -> 'a list -> 'b list = <fun>
val inc_all_by : int -> int list -> int list = <fun>
val filter : ('a -> bool) -> 'a list -> 'a list = <fun>
val removeABCD : char list -> char list = <fun>
val foldr : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b = <fun>
val foldl : ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a = <fun>
val f : int -> 'a -> int = <fun>
val length : 'a list -> int = <fun>
val foldl' : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b = <fun>
val length : 'a list -> int = <fun>
─( 15:41:32 )─< command 3 >──────────────────────────────{ counter: 0 }─
utop # length [1;2;3] ;;
- : int = 3
─( 15:41:39 )─< command 4 >──────────────────────────────{ counter: 0 }─
utop # foldr ;;
- : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b = <fun>
```

```
─( 15:41:43 )─< command 5 >──────────────────────────────{ counter: 0 }─
utop # #use "higher_order.ml";;
val implode : char list -> string = <fun>
val explode : string -> char list = <fun>
val map : ('a -> 'b) -> 'a list -> 'b list = <fun>
val inc_all_by : int -> int list -> int list = <fun>
val filter : ('a -> bool) -> 'a list -> 'a list = <fun>
val removeABCD : char list -> char list = <fun>
val foldr : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b = <fun>
val f : 'a -> int -> int = <fun>
val length : 'a list -> int = <fun>
val foldl : ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a = <fun>
val f : 'a list -> 'a -> 'a list = <fun>
val rev : 'a list -> 'a list = <fun>
val foldl' : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b = <fun>
val length : 'a list -> int = <fun>
─( 15:48:07 )─< command 6 >──────────────────────────────{ counter: 0 }─
utop # rev [1;2;3] ;;
- : int list = [3; 2; 1]
─( 15:54:34 )─< command 7 >──────────────────────────────{ counter: 0 }─
utop # List.fold_right ;;
- : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b = <fun>
─( 15:54:38 )─< command 8 >──────────────────────────────{ counter: 0 }─
utop # #use "higher_order.ml";;
val implode : char list -> string = <fun>
val explode : string -> char list = <fun>
val map : ('a -> 'b) -> 'a list -> 'b list = <fun>
val inc_all_by : int -> int list -> int list = <fun>
val filter : ('a -> bool) -> 'a list -> 'a list = <fun>
val removeABCD : char list -> char list = <fun>
val foldr : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b = <fun>
val f : 'a -> int -> int = <fun>
val length : 'a list -> int = <fun>
val foldl : ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a = <fun>
val foldl' : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b = <fun>
val length : 'a list -> int = <fun>
─( 16:04:20 )─< command 9 >──────────────────────────────{ counter: 0 }─
utop # #use "group_by_3.ml";;
val partition : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
─( 16:07:59 )─< command 10 >─────────────────────────────{ counter: 0 }─
utop # #use "group_by_3.ml";;
val even : int -> bool = <fun>
val ns : int list = [1; 2; 3; 4; 5; 6; 7; 8; 9; 10]
val partition : ('a -> bool) -> 'a list -> 'a list * 'a list =
```

```
  <fun>
─( 16:08:07 )─< command 11 >──────────────────────────────{ counter: 0 }─
utop # partition even ns ;;
- : int list * int list = ([2; 4; 6; 8; 10], [1; 3; 5; 7; 9])
─( 16:08:30 )─< command 12 >──────────────────────────────{ counter: 0 }─
utop # List.fold_left ;;
- : ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a = <fun>
─( 16:08:35 )─< command 13 >──────────────────────────────{ counter: 0 }─
utop # #use "group_by_3.ml";;
val even : int -> bool = <fun>
val ns : int list = [1; 2; 3; 4; 5; 6; 7; 8; 9; 10]
val partition_l : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
val partition_r : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
─( 16:10:08 )─< command 14 >──────────────────────────────{ counter: 0 }─
utop # partition_l even ns ;;
- : int list * int list = ([10; 8; 6; 4; 2], [9; 7; 5; 3; 1])
─( 16:10:22 )─< command 15 >──────────────────────────────{ counter: 0 }─
utop # #use "group_by_3.ml";;
val even : int -> bool = <fun>
val ns : int list = [1; 2; 3; 4; 5; 6; 7; 8; 9; 10]
val partition_l : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
val partition_r : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
─( 16:10:27 )─< command 16 >──────────────────────────────{ counter: 0 }─
utop # partition_l even ns ;;
- : int list * int list = ([2; 4; 6; 8; 10], [1; 3; 5; 7; 9])
─( 16:11:52 )─< command 17 >──────────────────────────────{ counter: 0 }─
utop # List.fold_left ;;
- : ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a = <fun>
─( 16:11:53 )─< command 18 >──────────────────────────────{ counter: 0 }─
utop # #use "group_by_3.ml";;
val even : int -> bool = <fun>
val ns : int list = [1; 2; 3; 4; 5; 6; 7; 8; 9; 10]
val partition_l : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
val partition_r : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
val group_by_3 : 'a list -> 'a list list * 'a list * int = <fun>
─( 16:19:53 )─< command 19 >──────────────────────────────{ counter: 0 }─
utop # group_by_3 [1;2;3;4;5;6;7;8;9;10];;
- : int list list * int list * int
= ([[7; 8; 9]; [4; 5; 6]; [1; 2; 3]], [10], 1)
```

```
─( 16:25:02 )─< command 20 >──────────────────────────────{ counter: 0 }─
utop # #use "group_by_3.ml";;
val even : int -> bool = <fun>
val ns : int list = [1; 2; 3; 4; 5; 6; 7; 8; 9; 10]
val partition_l : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
val partition_r : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
val group_by_3 : 'a list -> 'a list list = <fun>
─( 16:25:13 )─< command 21 >──────────────────────────────{ counter: 0 }─
utop # group_by_3 [1;2;3;4;5;6;7;8;9;10];;
- : int list list = [[1; 2; 3]; [4; 5; 6]; [7; 8; 9]; [10]]
─( 16:26:26 )─< command 22 >──────────────────────────────{ counter: 0 }─
utop # group_by_3 [1;2;3;4;5;6;7;8;9];;
- : int list list = [[1; 2; 3]; [4; 5; 6]; [7; 8; 9]]
─( 16:26:28 )─< command 23 >──────────────────────────────{ counter: 0 }─
utop # #use "group_by_3.ml";;
val even : int -> bool = <fun>
val ns : int list = [1; 2; 3; 4; 5; 6; 7; 8; 9; 10]
val partition_l : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
val partition_r : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
val group_by_3 : 'a list -> 'a list list * 'a list * int = <fun>
─( 16:49:57 )─< command 24 >──────────────────────────────{ counter: 0 }─
utop # group_by_3 [1;2;3;4;5;6;7;8;9];;
- : int list list * int list * int = ([[4; 5; 6]; [1; 2; 3]], [9; 8;
 7], 3)
─( 16:50:43 )─< command 25 >──────────────────────────────{ counter: 0 }─
utop # #use "group_by_3.ml";;
val even : int -> bool = <fun>
val ns : int list = [1; 2; 3; 4; 5; 6; 7; 8; 9; 10]
val partition_l : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
val partition_r : ('a -> bool) -> 'a list -> 'a list * 'a list =
  <fun>
val group_by_3 : 'a list -> 'a list list = <fun>
─( 16:50:43 )─< command 26 >──────────────────────────────{ counter: 0 }─
utop # group_by_3 [1;2;3;4;5;6;7;8;9];;
- : int list list = [[1; 2; 3]; [4; 5; 6]; [7; 8; 9]]
─( 16:51:12 )─< command 27 >──────────────────────────────{ counter: 0 }─
utop #
```

Arg	Array	ArrayLabels	Assert_failure	Bigarray	Buffer	Bytes	BytesLa