```
Last login: Wed Apr  4 12:29:29 on ttys009
carbon:$ pwd
/project/evw/Teaching/18_Spring_2041/carbon-repos/public-class-r
epo/Sample Programs/Sec_01_1-25pm/Search
carbon:$ cd ../../Sec_10_3-35pm/Search/
carbon:$ utop
```

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

```
Type #utop_help for help about using utop.

─( 15:41:53 )─< command 0 >──────────────────────────{ counter: 0 }─
utop # #use "search_exceptions.ml";;
val s : int list = [1; 3; -2; 5; -6]
val sum : int list -> int = <fun>
val show_list : ('a -> string) -> 'a list -> string = <fun>
exception FoundSubSet of int list
val subsetsum_exn_on_found : int list -> int list option =
  <fun>
exception KeepLooking
val subsetsum_exn_not_found : int list -> int list option =
  <fun>
val process_solution_exn :
  ('a list -> string) -> 'a list -> 'a list = <fun>
val subsetsum_exn : int list -> int list option = <fun>
val subsetsum_exn_continutation :
  int list -> (int list -> int list) -> int list option = <fun>
val subsetsum_exn_v1 : 'a -> 'b option = <fun>
val subsetsum_exn_first : 'a -> 'b option = <fun>
val subsetsum_exn_print_all : 'a -> 'b option = <fun>
val results : '_weak1 list ref = {contents = []}
val subsetsum_exn_save_all : 'a -> 'b option = <fun>
─( 15:41:53 )─< command 1 >──────────────────────────{ counter: 0 }─
utop # #use "search_exceptions.ml";;
val s : int list = [1; 3; -2; 5; -6]
val sum : int list -> int = <fun>
val show_list : ('a -> string) -> 'a list -> string = <fun>
exception FoundSubSet of int list
val subsetsum_exn_on_found : int list -> int list option =
  <fun>
exception KeepLooking
val subsetsum_exn_not_found : int list -> int list option =
```

```
  <fun>
val process_solution_exn :
  ('a list -> string) -> 'a list -> 'a list = <fun>
val subsetsum_exn : int list -> int list option = <fun>
val subsetsum_exn_continutation :
  int list -> (int list -> int list) -> int list option = <fun>
File "search_exceptions.ml", line 171, characters 2-28:
Error: Unbound value subsetsum_exn_continuation
Hint: Did you mean subsetsum_exn_continutation?
-( 15:41:59 )-< command 2 >-------------------------{ counter: 0 }-
utop # #use "search_exceptions.ml";;
val s : int list = [1; 3; -2; 5; -6]
val sum : int list -> int = <fun>
val show_list : ('a -> string) -> 'a list -> string = <fun>
exception FoundSubSet of int list
val subsetsum_exn_on_found : int list -> int list option =
  <fun>
exception KeepLooking
val subsetsum_exn_not_found : int list -> int list option =
  <fun>
val process_solution_exn :
  ('a list -> string) -> 'a list -> 'a list = <fun>
val subsetsum_exn : int list -> int list option = <fun>
val subsetsum_exn_continutation :
  int list -> (int list -> int list) -> int list option = <fun>
val subsetsum_exn_v1 : int list -> int list option = <fun>
val subsetsum_exn_first : 'a -> 'b option = <fun>
val subsetsum_exn_print_all : 'a -> 'b option = <fun>
val results : '_weak2 list ref = {contents = []}
val subsetsum_exn_save_all : 'a -> 'b option = <fun>
-( 15:48:39 )-< command 3 >-------------------------{ counter: 0 }-
utop # subsetsum_exn s ;;
Here is a solution: [ 1; 5; -6 ]
Do you like it ?
n
Here is a solution: [ 3; -2; 5; -6 ]
Do you like it ?
y
Thanks for playing...
- : int list option = Some [3; -2; 5; -6]
-( 15:48:56 )-< command 4 >-------------------------{ counter: 0 }-
utop # subsetsum_exn_v1 s ;;
Here is a solution: [ 1; 5; -6 ]
```

```
Do you like it ?
n
Here is a solution: [ 3; -2; 5; -6 ]
Do you like it ?
y
Thanks for playing...
- : int list option = Some [3; -2; 5; -6]
-( 15:49:11 )-< command 5 >-------------------------{ counter: 0 }-
utop # #use "search_exceptions.ml";;
val s : int list = [1; 3; -2; 5; -6]
val sum : int list -> int = <fun>
val show_list : ('a -> string) -> 'a list -> string = <fun>
exception FoundSubSet of int list
val subsetsum_exn_on_found : int list -> int list option =
  <fun>
exception KeepLooking
val subsetsum_exn_not_found : int list -> int list option =
  <fun>
val process_solution_exn :
  ('a list -> string) -> 'a list -> 'a list = <fun>
val subsetsum_exn : int list -> int list option = <fun>
val subsetsum_exn_continutation :
  int list -> (int list -> int list) -> int list option = <fun>
val subsetsum_exn_v1 : int list -> int list option = <fun>
val subsetsum_exn_first : int list -> int list option = <fun>
val subsetsum_exn_print_all : 'a -> 'b option = <fun>
val results : '_weak3 list ref = {contents = []}
val subsetsum_exn_save_all : 'a -> 'b option = <fun>
-( 15:49:35 )-< command 6 >-------------------------{ counter: 0 }-
utop # subsetsum_exn_first s ;;
- : int list option = Some [1; 5; -6]
-( 15:50:55 )-< command 7 >-------------------------{ counter: 0 }-
utop # #use "search_exceptions.ml";;
val s : int list = [1; 3; -2; 5; -6]
val sum : int list -> int = <fun>
val show_list : ('a -> string) -> 'a list -> string = <fun>
exception FoundSubSet of int list
val subsetsum_exn_on_found : int list -> int list option =
  <fun>
exception KeepLooking
val subsetsum_exn_not_found : int list -> int list option =
  <fun>
val process_solution_exn :
```

```
  ('a list -> string) -> 'a list -> 'a list = <fun>
val subsetsum_exn : int list -> int list option = <fun>
val subsetsum_exn_continutation :
  int list -> (int list -> int list) -> int list option = <fun>
val subsetsum_exn_v1 : int list -> int list option = <fun>
val subsetsum_exn_first : int list -> int list option = <fun>
File "search_exceptions.ml", line 192, characters 23-27:
Error: Unbound value show
-( 15:51:01 )-< command 8 >-------------------------{ counter: 0 }-
utop # #use "search_exceptions.ml";;
val s : int list = [1; 3; -2; 5; -6]
val sum : int list -> int = <fun>
val show_list : ('a -> string) -> 'a list -> string = <fun>
exception FoundSubSet of int list
val subsetsum_exn_on_found : int list -> int list option =
  <fun>
exception KeepLooking
val subsetsum_exn_not_found : int list -> int list option =
  <fun>
val process_solution_exn :
  ('a list -> string) -> 'a list -> 'a list = <fun>
val subsetsum_exn : int list -> int list option = <fun>
val subsetsum_exn_continutation :
  int list -> (int list -> int list) -> int list option = <fun>
val subsetsum_exn_v1 : int list -> int list option = <fun>
val subsetsum_exn_first : int list -> int list option = <fun>
File "search_exceptions.ml", line 192, characters 22-47:
Error: This function has type int list -> string
       It is applied to too many arguments;
       maybe you forgot a `;'.
-( 15:53:01 )-< command 9 >-------------------------{ counter: 0 }-
utop # #use "search_exceptions.ml";;
val s : int list = [1; 3; -2; 5; -6]
val sum : int list -> int = <fun>
val show_list : ('a -> string) -> 'a list -> string = <fun>
exception FoundSubSet of int list
val subsetsum_exn_on_found : int list -> int list option =
  <fun>
exception KeepLooking
val subsetsum_exn_not_found : int list -> int list option =
  <fun>
val process_solution_exn :
  ('a list -> string) -> 'a list -> 'a list = <fun>
```

```
val subsetsum_exn : int list -> int list option = <fun>
val subsetsum_exn_continutation :
  int list -> (int list -> int list) -> int list option = <fun>
val subsetsum_exn_v1 : int list -> int list option = <fun>
val subsetsum_exn_first : int list -> int list option = <fun>
val subsetsum_exn_print_all : int list -> int list option =
  <fun>
val results : '_weak4 list ref = {contents = []}
val subsetsum_exn_save_all : 'a -> 'b option = <fun>
-( 15:53:20 )-< command 10 >-------------------------{ counter: 0 }-
utop # subsetsum_exn_print_all s ;;
Here is a solution: [ 1; 5; -6 ]
Here is a solution: [ 3; -2; 5; -6 ]
- : int list option = None
-( 15:53:28 )-< command 11 >-------------------------{ counter: 0 }-
utop # #use "search_cps.ml";;
val show_list : ('a -> string) -> 'a list -> string = <fun>
val sum : int list -> int = <fun>
val process_solution_cps_v1 :
  ('a -> string) -> 'a -> (unit -> 'b) -> (unit -> 'b) -> 'b =
  <fun>
val try_subset_cps_v1 :
  int list -> int list -> (unit -> 'a) -> (unit -> 'a) -> 'a =
  <fun>
val subsetsum_cps_v1 : int list -> unit = <fun>
val process_solution_cps_v2 :
  ('a -> string) -> 'a -> 'b -> 'c -> unit = <fun>
val try_subset_cps_v2 :
  int list -> int list -> 'a -> (unit -> unit) -> unit = <fun>
val subsetsum_cps_v2 : int list -> unit = <fun>
-( 15:53:35 )-< command 12 >-------------------------{ counter: 0 }-
utop # subsetsum_cps_v1 s ;;
Oh no, no solution.
- : unit = ()
-( 15:59:58 )-< command 13 >-------------------------{ counter: 0 }-
utop # #use "search_cps.ml";;
val show_list : ('a -> string) -> 'a list -> string = <fun>
val sum : int list -> int = <fun>
val process_solution_cps_v1 :
  ('a -> string) -> 'a -> (unit -> 'b) -> (unit -> 'b) -> 'b =
  <fun>
val try_subset_cps_v1 :
  int list -> int list -> (unit -> 'a) -> (unit -> 'a) -> 'a =
```

```
  <fun>
val subsetsum_cps_v1 : int list -> unit = <fun>
val process_solution_cps_v2 :
  ('a -> string) -> 'a -> 'b -> 'c -> unit = <fun>
val try_subset_cps_v2 :
  int list -> int list -> 'a -> (unit -> unit) -> unit = <fun>
val subsetsum_cps_v2 : int list -> unit = <fun>
─( 16:00:10 )─< command 14 >──────────────────────────{ counter: 0 }─
utop # subsetsum_cps_v1 s ;;
Here is a solution:
[ 1; 5; -6 ]
Do you like it?
n
Here is a solution:
[ 3; -2; 5; -6 ]
Do you like it?
y
Yeah, found a solution.
- : unit = ()
─( 16:02:28 )─< command 15 >──────────────────────────{ counter: 0 }─
utop # #use "wolf.ml";;
File "wolf.ml", line 87, characters 35-36:
Error: Syntax error: operator expected.
─( 16:02:33 )─< command 16 >──────────────────────────{ counter: 0 }─
utop # #use "wolf.ml";;
val is_not_elem : 'a list -> 'a -> bool = <fun>
type loc = L | R
type state = loc * loc * loc * loc
val ok_state : state -> bool = <fun>
val final : loc * loc * loc * loc -> bool = <fun>
val other_side : loc -> loc = <fun>
val moves : state -> state list = <fun>
File "wolf.ml", line 83, characters 6-270:
Warning 8: this pattern-matching is not exhaustive.
Here is an example of a case that is not matched:
_::_::_::_
val crossing_v1 : unit -> state list option = <fun>
exception FoundPath of (loc * loc * loc * loc) list
val crossing_v2 : unit -> unit = <fun>
val crossing_many_possible_moves : unit -> unit = <fun>
val crossing_many_possible_moves' : unit -> unit = <fun>
exception KeepLooking
val process_solution_exn : ('a -> string) -> 'a -> 'a option =
```

```
  <fun>
val show_list : ('a -> string) -> 'a list -> string = <fun>
val show_loc : 'a -> string = <fun>
val show_state : 'a * 'b * 'c * 'd -> string = <fun>
val show_path :
  ('_weak5 * '_weak6 * '_weak7 * '_weak8) list -> string =
  <fun>
File "wolf.ml", line 169, characters 22-26:
Error: This variant expression is expected to have type unit
       The constructor None does not belong to type unit
-( 16:25:00 )-< command 17 >-------------------------{ counter: 0 }-
utop # #use "wolf.ml";;
val is_not_elem : 'a list -> 'a -> bool = <fun>
type loc = L | R
type state = loc * loc * loc * loc
val ok_state : state -> bool = <fun>
val final : loc * loc * loc * loc -> bool = <fun>
val other_side : loc -> loc = <fun>
val moves : state -> state list = <fun>
File "wolf.ml", line 83, characters 6-270:
Warning 8: this pattern-matching is not exhaustive.
Here is an example of a case that is not matched:
_::_::_::_
val crossing_v1 : unit -> state list option = <fun>
exception FoundPath of (loc * loc * loc * loc) list
val crossing_v2 : unit -> unit = <fun>
val crossing_many_possible_moves : unit -> unit = <fun>
val crossing_many_possible_moves' : unit -> unit = <fun>
exception KeepLooking
val process_solution_exn : ('a -> string) -> 'a -> 'a option =
  <fun>
val show_list : ('a -> string) -> 'a list -> string = <fun>
val show_loc : 'a -> string = <fun>
val show_state : 'a * 'b * 'c * 'd -> string = <fun>
val show_path :
  ('_weak9 * '_weak10 * '_weak11 * '_weak12) list -> string =
  <fun>
-( 16:25:36 )-< command 18 >-------------------------{ counter: 0 }-
utop # crossing_v1 () ;;
- : state list option =
Some
 [(L, L, L, L); (R, L, R, L); (L, L, R, L); (R, R, R, L);
  (L, R, L, L); (R, R, L, R); (L, R, L, R); (R, R, R, R)]
```