Stdlib
ArrayLabels.append : 'a array -> 'a array -> 'a array
ArrayLabels.blit : src:'a array -> src_pos:int -> dst:'a array -> dst_pos:int -> len:int -> unit
ArrayLabels.combine : 'a array -> 'b array -> ('a * 'b) array
ArrayLabels.concat : 'a array list -> 'a array
ArrayLabels.copy : 'a array -> 'a array
ArrayLabels.create_matrix : dimx:int -> dimy:int -> 'a -> 'a array array
ArrayLabels.exists : f:('a -> bool) -> 'a array -> bool
ArrayLabels.exists2 : f:('a -> 'b -> bool) -> 'a array -> 'b array -> bool
ArrayLabels.fast_sort : cmp:('a -> 'a -> int) -> 'a array -> unit
ArrayLabels.fill : 'a array -> pos:int -> len:int -> 'a -> unit
ArrayLabels.find_map : f:('a -> 'b option) -> 'a array -> 'b option
ArrayLabels.find_opt : f:('a -> bool) -> 'a array -> 'a option
ArrayLabels.fold_left : f:('a -> 'b -> 'a) -> init:'a -> 'b array -> 'a
ArrayLabels.fold_left_map : f:('a -> 'b -> 'a * 'c) -> init:'a -> 'b array -> 'a * 'c array
ArrayLabels.fold_right : f:('b -> 'a -> 'a) -> 'b array -> init:'a -> 'a
ArrayLabels.for_all : f:('a -> bool) -> 'a array -> bool
ArrayLabels.for_all2 : f:('a -> 'b -> bool) -> 'a array -> 'b array -> bool
ArrayLabels.init : int -> f:(int -> 'a) -> 'a array
ArrayLabels.iter : f:('a -> unit) -> 'a array -> unit
ArrayLabels.iter2 : f:('a -> 'b -> unit) -> 'a array -> 'b array -> unit
ArrayLabels.iteri : f:(int -> 'a -> unit) -> 'a array -> unit
ArrayLabels.make_float : int -> float array
ArrayLabels.make_matrix : dimx:int -> dimy:int -> 'a -> 'a array array
ArrayLabels.map : f:('a -> 'b) -> 'a array -> 'b array
ArrayLabels.map2 : f:('a -> 'b -> 'c) -> 'a array -> 'b array -> 'c array
ArrayLabels.mapi : f:(int -> 'a -> 'b) -> 'a array -> 'b array
ArrayLabels.mem : 'a -> set:'a array -> bool
ArrayLabels.memq : 'a -> set:'a array -> bool
ArrayLabels.of_list : 'a list -> 'a array
ArrayLabels.of_seq : 'a Seq.t -> 'a array
ArrayLabels.sort : cmp:('a -> 'a -> int) -> 'a array -> unit
ArrayLabels.split : ('a * 'b) array -> 'a array * 'b array
ArrayLabels.stable_sort : cmp:('a -> 'a -> int) -> 'a array -> unit
ArrayLabels.sub : 'a array -> pos:int -> len:int -> 'a array

Stdlib
ArrayLabels.to_list : 'a array -> 'a list
ArrayLabels.to_seq : 'a array -> 'a Seq.t
ArrayLabels.to_seqi : 'a array -> (int * 'a) Seq.t
Array.append : 'a array -> 'a array -> 'a array
Array.blit : 'a array -> int -> 'a array -> int -> int -> unit
Array.combine : 'a array -> 'b array -> ('a * 'b) array
Array.concat : 'a array list -> 'a array
Array.copy : 'a array -> 'a array
Array.create_matrix : int -> int -> 'a -> 'a array array
Array.exists : ('a -> bool) -> 'a array -> bool
Array.exists2 : ('a -> 'b -> bool) -> 'a array -> 'b array -> bool
Array.fast_sort : ('a -> 'a -> int) -> 'a array -> unit
Array.fill : 'a array -> int -> int -> 'a -> unit
Array.find_map : ('a -> 'b option) -> 'a array -> 'b option
Array.find_opt : ('a -> bool) -> 'a array -> 'a option
Array.fold_left : ('a -> 'b -> 'a) -> 'a -> 'b array -> 'a
Array.fold_left_map : ('a -> 'b -> 'a * 'c) -> 'a -> 'b array -> 'a * 'c array
Array.fold_right : ('b -> 'a -> 'a) -> 'b array -> 'a -> 'a
Array.for_all : ('a -> bool) -> 'a array -> bool
Array.for_all2 : ('a -> 'b -> bool) -> 'a array -> 'b array -> bool
Array.init : int -> (int -> 'a) -> 'a array
Array.iter : ('a -> unit) -> 'a array -> unit
Array.iter2 : ('a -> 'b -> unit) -> 'a array -> 'b array -> unit
Array.iteri : (int -> 'a -> unit) -> 'a array -> unit
Array.make_float : int -> float array
Array.make_matrix : int -> int -> 'a -> 'a array array
Array.map : ('a -> 'b) -> 'a array -> 'b array
Array.map2 : ('a -> 'b -> 'c) -> 'a array -> 'b array -> 'c array
Array.mapi : (int -> 'a -> 'b) -> 'a array -> 'b array
Array.mem : 'a -> 'a array -> bool
Array.memq : 'a -> 'a array -> bool
Array.of_list : 'a list -> 'a array
Array.of_seq : 'a Seq.t -> 'a array
Array.sort : ('a -> 'a -> int) -> 'a array -> unit

Stdlib
Array.split : ('a * 'b) array -> 'a array * 'b array
Array.stable_sort : ('a -> 'a -> int) -> 'a array -> unit
Array.sub : 'a array -> int -> int -> 'a array
Array.to_list : 'a array -> 'a list
Array.to_seq : 'a array -> 'a Seq.t
Array.to_seqi : 'a array -> (int * 'a) Seq.t
ListLabels.append : 'a list -> 'a list -> 'a list
ListLabels.assoc : 'a -> ('a * 'b) list -> 'b
ListLabels.assoc_opt : 'a -> ('a * 'b) list -> 'b option
ListLabels.assq : 'a -> ('a * 'b) list -> 'b
ListLabels.assq_opt : 'a -> ('a * 'b) list -> 'b option
ListLabels.combine : 'a list -> 'b list -> ('a * 'b) list
ListLabels.compare : cmp:('a -> 'a -> int) -> 'a list -> 'a list -> int
ListLabels.compare_length_with : 'a list -> len:int -> int
ListLabels.compare_lengths : 'a list -> 'b list -> int
ListLabels.concat : 'a list list -> 'a list
ListLabels.concat_map : f:('a -> 'b list) -> 'a list -> 'b list
ListLabels.cons : 'a -> 'a list -> 'a list
ListLabels.equal : eq:('a -> 'a -> bool) -> 'a list -> 'a list -> bool
ListLabels.exists : f:('a -> bool) -> 'a list -> bool
ListLabels.exists2 : f:('a -> 'b -> bool) -> 'a list -> 'b list -> bool
ListLabels.fast_sort : cmp:('a -> 'a -> int) -> 'a list -> 'a list
ListLabels.filter : f:('a -> bool) -> 'a list -> 'a list
ListLabels.filter_map : f:('a -> 'b option) -> 'a list -> 'b list
ListLabels.filteri : f:(int -> 'a -> bool) -> 'a list -> 'a list
ListLabels.find : f:('a -> bool) -> 'a list -> 'a
ListLabels.find_all : f:('a -> bool) -> 'a list -> 'a list
ListLabels.find_map : f:('a -> 'b option) -> 'a list -> 'b option
ListLabels.find_opt : f:('a -> bool) -> 'a list -> 'a option
ListLabels.flatten : 'a list list -> 'a list
ListLabels.fold_left : f:('a -> 'b -> 'a) -> init:'a -> 'b list -> 'a
ListLabels.fold_left2 : f:('a -> 'b -> 'c -> 'a) -> init:'a -> 'b list -> 'c list -> 'a
ListLabels.fold_left_map : f:('a -> 'b -> 'a * 'c) -> init:'a -> 'b list -> 'a * 'c list
ListLabels.fold_right : f:('a -> 'b -> 'b) -> 'a list -> init:'b -> 'b

Stdlib
ListLabels.fold_right2 : f:('a -> 'b -> 'c -> 'c) -> 'a list -> 'b list -> init:'c -> 'c
ListLabels.for_all : f:('a -> bool) -> 'a list -> bool
ListLabels.for_all2 : f:('a -> 'b -> bool) -> 'a list -> 'b list -> bool
ListLabels.hd : 'a list -> 'a
ListLabels.init : len:int -> f:(int -> 'a) -> 'a list
ListLabels.iter : f:('a -> unit) -> 'a list -> unit
ListLabels.iter2 : f:('a -> 'b -> unit) -> 'a list -> 'b list -> unit
ListLabels.iteri : f:(int -> 'a -> unit) -> 'a list -> unit
ListLabels.length : 'a list -> int
ListLabels.map : f:('a -> 'b) -> 'a list -> 'b list
ListLabels.map2 : f:('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list
ListLabels.mapi : f:(int -> 'a -> 'b) -> 'a list -> 'b list
ListLabels.mem : 'a -> set:'a list -> bool
ListLabels.mem_assoc : 'a -> map:('a * 'b) list -> bool
ListLabels.mem_assq : 'a -> map:('a * 'b) list -> bool
ListLabels.memq : 'a -> set:'a list -> bool
ListLabels.merge : cmp:('a -> 'a -> int) -> 'a list -> 'a list -> 'a list
ListLabels.nth : 'a list -> int -> 'a
ListLabels.nth_opt : 'a list -> int -> 'a option
ListLabels.of_seq : 'a Seq.t -> 'a list
ListLabels.partition : f:('a -> bool) -> 'a list -> 'a list * 'a list
ListLabels.partition_map : f:('a -> ('b, 'c) Either.t) -> 'a list -> 'b list * 'c list
ListLabels.remove_assoc : 'a -> ('a * 'b) list -> ('a * 'b) list
ListLabels.remove_assq : 'a -> ('a * 'b) list -> ('a * 'b) list
ListLabels.rev : 'a list -> 'a list
ListLabels.rev_append : 'a list -> 'a list -> 'a list
ListLabels.rev_map : f:('a -> 'b) -> 'a list -> 'b list
ListLabels.rev_map2 : f:('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list
ListLabels.sort : cmp:('a -> 'a -> int) -> 'a list -> 'a list
ListLabels.sort_uniq : cmp:('a -> 'a -> int) -> 'a list -> 'a list
ListLabels.split : ('a * 'b) list -> 'a list * 'b list
ListLabels.stable_sort : cmp:('a -> 'a -> int) -> 'a list -> 'a list
ListLabels.tl : 'a list -> 'a list
ListLabels.to_seq : 'a list -> 'a Seq.t

Stdlib
List.append : 'a list -> 'a list -> 'a list
List.assoc : 'a -> ('a * 'b) list -> 'b
List.assoc_opt : 'a -> ('a * 'b) list -> 'b option
List.assq : 'a -> ('a * 'b) list -> 'b
List.assq_opt : 'a -> ('a * 'b) list -> 'b option
List.combine : 'a list -> 'b list -> ('a * 'b) list
List.compare : ('a -> 'a -> int) -> 'a list -> 'a list -> int
List.compare_length_with : 'a list -> int -> int
List.compare_lengths : 'a list -> 'b list -> int
List.concat : 'a list list -> 'a list
List.concat_map : ('a -> 'b list) -> 'a list -> 'b list
List.cons : 'a -> 'a list -> 'a list
List.equal : ('a -> 'a -> bool) -> 'a list -> 'a list -> bool
List.exists : ('a -> bool) -> 'a list -> bool
List.exists2 : ('a -> 'b -> bool) -> 'a list -> 'b list -> bool
List.fast_sort : ('a -> 'a -> int) -> 'a list -> 'a list
List.filter : ('a -> bool) -> 'a list -> 'a list
List.filter_map : ('a -> 'b option) -> 'a list -> 'b list
List.filteri : (int -> 'a -> bool) -> 'a list -> 'a list
List.find : ('a -> bool) -> 'a list -> 'a
List.find_all : ('a -> bool) -> 'a list -> 'a list
List.find_map : ('a -> 'b option) -> 'a list -> 'b option
List.find_opt : ('a -> bool) -> 'a list -> 'a option
List.flatten : 'a list list -> 'a list
List.fold_left : ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a
List.fold_left2 : ('a -> 'b -> 'c -> 'a) -> 'a -> 'b list -> 'c list -> 'a
List.fold_left_map : ('a -> 'b -> 'a * 'c) -> 'a -> 'b list -> 'a * 'c list
List.fold_right : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b
List.fold_right2 : ('a -> 'b -> 'c -> 'c) -> 'a list -> 'b list -> 'c -> 'c
List.for_all : ('a -> bool) -> 'a list -> bool
List.for_all2 : ('a -> 'b -> bool) -> 'a list -> 'b list -> bool
List.hd : 'a list -> 'a
List.init : int -> (int -> 'a) -> 'a list
List.iter : ('a -> unit) -> 'a list -> unit

Stdlib
List.iter2 : ('a -> 'b -> unit) -> 'a list -> 'b list -> unit
List.iteri : (int -> 'a -> unit) -> 'a list -> unit
List.length : 'a list -> int
List.map : ('a -> 'b) -> 'a list -> 'b list
List.map2 : ('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list
List.mapi : (int -> 'a -> 'b) -> 'a list -> 'b list
List.mem : 'a -> 'a list -> bool
List.mem_assoc : 'a -> ('a * 'b) list -> bool
List.mem_assq : 'a -> ('a * 'b) list -> bool
List.memq : 'a -> 'a list -> bool
List.merge : ('a -> 'a -> int) -> 'a list -> 'a list -> 'a list
List.nth : 'a list -> int -> 'a
List.nth_opt : 'a list -> int -> 'a option
List.of_seq : 'a Seq.t -> 'a list
List.partition : ('a -> bool) -> 'a list -> 'a list * 'a list
List.partition_map : ('a -> ('b, 'c) Either.t) -> 'a list -> 'b list * 'c list
List.remove_assoc : 'a -> ('a * 'b) list -> ('a * 'b) list
List.remove_assq : 'a -> ('a * 'b) list -> ('a * 'b) list
List.rev : 'a list -> 'a list
List.rev_append : 'a list -> 'a list -> 'a list
List.rev_map : ('a -> 'b) -> 'a list -> 'b list
List.rev_map2 : ('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list
List.sort : ('a -> 'a -> int) -> 'a list -> 'a list
List.sort_uniq : ('a -> 'a -> int) -> 'a list -> 'a list
List.split : ('a * 'b) list -> 'a list * 'b list
List.stable_sort : ('a -> 'a -> int) -> 'a list -> 'a list
List.tl : 'a list -> 'a list
List.to_seq : 'a list -> 'a Seq.t
Map.add : key -> 'a -> 'a t -> 'a t
Map.add_seq : (key * 'a) Seq.t -> 'a t -> 'a t
Map.bindings : 'a t -> (key * 'a) list
Map.cardinal : 'a t -> int
Map.choose : 'a t -> key * 'a
Map.choose_opt : 'a t -> (key * 'a) option

Stdlib
Map.compare : ('a -> 'a -> int) -> 'a t -> 'a t -> int
Map.empty : 'a t
Map.equal : ('a -> 'a -> bool) -> 'a t -> 'a t -> bool
Map.exists : (key -> 'a -> bool) -> 'a t -> bool
Map.filter : (key -> 'a -> bool) -> 'a t -> 'a t
Map.filter_map : (key -> 'a -> 'b option) -> 'a t -> 'b t
Map.find : key -> 'a t -> 'a
Map.find_first : (key -> bool) -> 'a t -> key * 'a
Map.find_first_opt : (key -> bool) -> 'a t -> (key * 'a) option
Map.find_last : (key -> bool) -> 'a t -> key * 'a
Map.find_last_opt : (key -> bool) -> 'a t -> (key * 'a) option
Map.find_opt : key -> 'a t -> 'a option
Map.fold : (key -> 'a -> 'b -> 'b) -> 'a t -> 'b -> 'b
Map.for_all : (key -> 'a -> bool) -> 'a t -> bool
Map.is_empty : 'a t -> bool
Map.iter : (key -> 'a -> unit) -> 'a t -> unit
Map.map : ('a -> 'b) -> 'a t -> 'b t
Map.mapi : (key -> 'a -> 'b) -> 'a t -> 'b t
Map.max_binding : 'a t -> key * 'a
Map.max_binding_opt : 'a t -> (key * 'a) option
Map.mem : key -> 'a t -> bool
Map.merge : (key -> 'a option -> 'b option -> 'c option) -> 'a t -> 'b t -> 'c t
Map.min_binding : 'a t -> key * 'a
Map.min_binding_opt : 'a t -> (key * 'a) option
Map.of_seq : (key * 'a) Seq.t -> 'a t
Map.partition : (key -> 'a -> bool) -> 'a t -> 'a t * 'a t
Map.remove : key -> 'a t -> 'a t
Map.singleton : key -> 'a -> 'a t
Map.split : key -> 'a t -> 'a t * 'a option * 'a t
Map.to_rev_seq : 'a t -> (key * 'a) Seq.t
Map.to_seq : 'a t -> (key * 'a) Seq.t
Map.to_seq_from : key -> 'a t -> (key * 'a) Seq.t
Map.union : (key -> 'a -> 'a -> 'a option) -> 'a t -> 'a t -> 'a t
Map.update : key -> ('a option -> 'a option) -> 'a t -> 'a t

Stdlib
Option.bind : 'a option -> ('a -> 'b option) -> 'b option
Option.compare : ('a -> 'a -> int) -> 'a option -> 'a option -> int
Option.equal : ('a -> 'a -> bool) -> 'a option -> 'a option -> bool
Option.fold : none:'a -> some:('b -> 'a) -> 'b option -> 'a
Option.get : 'a option -> 'a
Option.is_none : 'a option -> bool
Option.is_some : 'a option -> bool
Option.iter : ('a -> unit) -> 'a option -> unit
Option.join : 'a option option -> 'a option
Option.map : ('a -> 'b) -> 'a option -> 'b option
Option.none : 'a option
Option.some : 'a -> 'a option
Option.to_list : 'a option -> 'a list
Option.to_result : none:'e -> 'a option -> ('a, 'e) result
Option.to_seq : 'a option -> 'a Seq.t
Option.value : 'a option -> default:'a -> 'a
Printf.bprintf : Buffer.t -> ('a, Buffer.t, unit) format -> 'a
Printf.eprintf : ('a, out_channel, unit) format -> 'a
Printf.fprintf : out_channel -> ('a, out_channel, unit) format -> 'a
Printf.ibprintf : Buffer.t -> ('a, Buffer.t, unit) format -> 'a
Printf.ifprintf : 'b -> ('a, 'b, 'c, unit) format4 -> 'a
Printf.ikbprintf : (Buffer.t -> 'd) -> Buffer.t -> ('a, Buffer.t, unit, 'd) format4 -> 'a
Printf.ikfprintf : ('b -> 'd) -> 'b -> ('a, 'b, 'c, 'd) format4 -> 'a
Printf.kbprintf : (Buffer.t -> 'd) -> Buffer.t -> ('a, Buffer.t, unit, 'd) format4 -> 'a
Printf.kfprintf : (out_channel -> 'd) -> out_channel -> ('a, out_channel, unit, 'd) format4 -> 'a
Printf.kprintf : (string -> 'b) -> ('a, unit, string, 'b) format4 -> 'a
Printf.ksprintf : (string -> 'd) -> ('a, unit, string, 'd) format4 -> 'a
Printf.printf : ('a, out_channel, unit) format -> 'a
Printf.sprintf : ('a, unit, string) format -> 'a
Result.bind : ('a, 'e) result -> ('a -> ('b, 'e) result) -> ('b, 'e) result
Result.compare : ok:('a -> 'a -> int) -> error:('e -> 'e -> int) -> ('a, 'e) result -> ('a, 'e) result -> int
Result.equal : ok:('a -> 'a -> bool) -> error:('e -> 'e -> bool) -> ('a, 'e) result -> ('a, 'e) result -> bool
Result.error : 'e -> ('a, 'e) result
Result.fold : ok:('a -> 'c) -> error:('e -> 'c) -> ('a, 'e) result -> 'c

Stdlib
Result.get_error : ('a, 'e) result -> 'e
Result.get_ok : ('a, 'e) result -> 'a
Result.is_error : ('a, 'e) result -> bool
Result.is_ok : ('a, 'e) result -> bool
Result.iter : ('a -> unit) -> ('a, 'e) result -> unit
Result.iter_error : ('e -> unit) -> ('a, 'e) result -> unit
Result.join : (('a, 'e) result, 'e) result -> ('a, 'e) result
Result.map : ('a -> 'b) -> ('a, 'e) result -> ('b, 'e) result
Result.map_error : ('e -> 'f) -> ('a, 'e) result -> ('a, 'f) result
Result.ok : 'a -> ('a, 'e) result
Result.to_list : ('a, 'e) result -> 'a list
Result.to_option : ('a, 'e) result -> 'a option
Result.to_seq : ('a, 'e) result -> 'a Seq.t
Result.value : ('a, 'e) result -> default:'a -> 'a
Seq.append : 'a t -> 'a t -> 'a t
Seq.compare : ('a -> 'b -> int) -> 'a t -> 'b t -> int
Seq.concat : 'a t t -> 'a t
Seq.concat_map : ('a -> 'b t) -> 'a t -> 'b t
Seq.cons : 'a -> 'a t -> 'a t
Seq.cycle : 'a t -> 'a t
Seq.drop : int -> 'a t -> 'a t
Seq.drop_while : ('a -> bool) -> 'a t -> 'a t
Seq.empty : 'a t
Seq.equal : ('a -> 'b -> bool) -> 'a t -> 'b t -> bool
Seq.exists : ('a -> bool) -> 'a t -> bool
Seq.exists2 : ('a -> 'b -> bool) -> 'a t -> 'b t -> bool
Seq.filter : ('a -> bool) -> 'a t -> 'a t
Seq.filter_map : ('a -> 'b option) -> 'a t -> 'b t
Seq.find : ('a -> bool) -> 'a t -> 'a option
Seq.find_map : ('a -> 'b option) -> 'a t -> 'b option
Seq.flat_map : ('a -> 'b t) -> 'a t -> 'b t
Seq.fold_left : ('a -> 'b -> 'a) -> 'a -> 'b t -> 'a
Seq.fold_left2 : ('a -> 'b -> 'c -> 'a) -> 'a -> 'b t -> 'c t -> 'a
Seq.fold_lefti : ('b -> int -> 'a -> 'b) -> 'b -> 'a t -> 'b

Stdlib
Seq.for_all : ('a -> bool) -> 'a t -> bool
Seq.for_all2 : ('a -> 'b -> bool) -> 'a t -> 'b t -> bool
Seq.forever : (unit -> 'a) -> 'a t
Seq.group : ('a -> 'a -> bool) -> 'a t -> 'a t t
Seq.init : int -> (int -> 'a) -> 'a t
Seq.interleave : 'a t -> 'a t -> 'a t
Seq.ints : int -> int t
Seq.is_empty : 'a t -> bool
Seq.iter : ('a -> unit) -> 'a t -> unit
Seq.iter2 : ('a -> 'b -> unit) -> 'a t -> 'b t -> unit
Seq.iterate : ('a -> 'a) -> 'a -> 'a t
Seq.iteri : (int -> 'a -> unit) -> 'a t -> unit
Seq.length : 'a t -> int
Seq.map : ('a -> 'b) -> 'a t -> 'b t
Seq.map2 : ('a -> 'b -> 'c) -> 'a t -> 'b t -> 'c t
Seq.map_product : ('a -> 'b -> 'c) -> 'a t -> 'b t -> 'c t
Seq.mapi : (int -> 'a -> 'b) -> 'a t -> 'b t
Seq.memoize : 'a t -> 'a t
Seq.of_dispenser : (unit -> 'a option) -> 'a t
Seq.once : 'a t -> 'a t
Seq.partition : ('a -> bool) -> 'a t -> 'a t * 'a t
Seq.partition_map : ('a -> ('b, 'c) Either.t) -> 'a t -> 'b t * 'c t
Seq.product : 'a t -> 'b t -> ('a * 'b) t
Seq.repeat : 'a -> 'a t
Seq.return : 'a -> 'a t
Seq.scan : ('b -> 'a -> 'b) -> 'b -> 'a t -> 'b t
Seq.sorted_merge : ('a -> 'a -> int) -> 'a t -> 'a t -> 'a t
Seq.split : ('a * 'b) t -> 'a t * 'b t
Seq.take : int -> 'a t -> 'a t
Seq.take_while : ('a -> bool) -> 'a t -> 'a t
Seq.to_dispenser : 'a t -> unit -> 'a option
Seq.transpose : 'a t t -> 'a t t
Seq.uncons : 'a t -> ('a * 'a t) option
Seq.unfold : ('b -> ('a * 'b) option) -> 'b -> 'a t

Stdlib
Seq.unzip : ('a * 'b) t -> 'a t * 'b t
Seq.zip : 'a t -> 'b t -> ('a * 'b) t
Set.add : elt -> t -> t
Set.add_seq : elt Seq.t -> t -> t
Set.cardinal : t -> int
Set.choose : t -> elt
Set.choose_opt : t -> elt option
Set.compare : t -> t -> int
Set.diff : t -> t -> t
Set.disjoint : t -> t -> bool
Set.elements : t -> elt list
Set.empty : t
Set.equal : t -> t -> bool
Set.exists : (elt -> bool) -> t -> bool
Set.filter : (elt -> bool) -> t -> t
Set.filter_map : (elt -> elt option) -> t -> t
Set.find : elt -> t -> elt
Set.find_first : (elt -> bool) -> t -> elt
Set.find_first_opt : (elt -> bool) -> t -> elt option
Set.find_last : (elt -> bool) -> t -> elt
Set.find_last_opt : (elt -> bool) -> t -> elt option
Set.find_opt : elt -> t -> elt option
Set.fold : (elt -> 'a -> 'a) -> t -> 'a -> 'a
Set.for_all : (elt -> bool) -> t -> bool
Set.inter : t -> t -> t
Set.is_empty : t -> bool
Set.iter : (elt -> unit) -> t -> unit
Set.map : (elt -> elt) -> t -> t
Set.max_elt : t -> elt
Set.max_elt_opt : t -> elt option
Set.mem : elt -> t -> bool
Set.min_elt : t -> elt
Set.min_elt_opt : t -> elt option
Set.of_list : elt list -> t

Stdlib
Set.of_seq : elt Seq.t -> t
Set.partition : (elt -> bool) -> t -> t * t
Set.remove : elt -> t -> t
Set.singleton : elt -> t
Set.split : elt -> t -> t * bool * t
Set.subset : t -> t -> bool
Set.to_rev_seq : t -> elt Seq.t
Set.to_seq : t -> elt Seq.t
Set.to_seq_from : elt -> t -> elt Seq.t
Set.union : t -> t -> t
StringLabels.make : int -> char -> string
StringLabels.init : int -> f:(int -> char) -> string
StringLabels.empty : string
StringLabels.of_bytes : bytes -> string
StringLabels.to_bytes : string -> bytes
StringLabels.concat : sep:string -> string list -> string
StringLabels.cat : string -> string -> string
StringLabels.equal : t -> t -> bool
StringLabels.compare : t -> t -> int
StringLabels.starts_with : prefix:string -> string -> bool
StringLabels.ends_with : suffix:string -> string -> bool
StringLabels.contains_from : string -> int -> char -> bool
StringLabels.rcontains_from : string -> int -> char -> bool
StringLabels.contains : string -> char -> bool
StringLabels.sub : string -> pos:int -> len:int -> string
StringLabels.split_on_char : sep:char -> string -> string list
StringLabels.map : f:(char -> char) -> string -> string
StringLabels.mapi : f:(int -> char -> char) -> string -> string
StringLabels.fold_left : f:('a -> char -> 'a) -> init:'a -> string -> 'a
StringLabels.fold_right : f:(char -> 'a -> 'a) -> string -> init:'a -> 'a
StringLabels.for_all : f:(char -> bool) -> string -> bool
StringLabels.exists : f:(char -> bool) -> string -> bool
StringLabels.trim : string -> string
StringLabels.escaped : string -> string

Stdlib
StringLabels.uppercase_ascii : string -> string
StringLabels.lowercase_ascii : string -> string
StringLabels.capitalize_ascii : string -> string
StringLabels.uncapitalize_ascii : string -> string
StringLabels.iter : f:(char -> unit) -> string -> unit
StringLabels.iteri : f:(int -> char -> unit) -> string -> unit
StringLabels.index_from : string -> int -> char -> int
StringLabels.index_from_opt : string -> int -> char -> int option
StringLabels.rindex_from : string -> int -> char -> int
StringLabels.rindex_from_opt : string -> int -> char -> int option
StringLabels.index : string -> char -> int
StringLabels.index_opt : string -> char -> int option
StringLabels.rindex : string -> char -> int
StringLabels.rindex_opt : string -> char -> int option
StringLabels.to_seq : t -> char Seq.t
StringLabels.to_seqi : t -> (int * char) Seq.t
StringLabels.of_seq : char Seq.t -> t
StringLabels.get_utf_8_uchar : t -> int -> Uchar.utf_decode
StringLabels.is_valid_utf_8 : t -> bool
StringLabels.get_utf_16be_uchar : t -> int -> Uchar.utf_decode
StringLabels.is_valid_utf_16be : t -> bool
StringLabels.get_utf_16le_uchar : t -> int -> Uchar.utf_decode
StringLabels.is_valid_utf_16le : t -> bool
StringLabels.blit : src:string -> src_pos:int -> dst:bytes -> dst_pos:int -> len:int -> unit
StringLabels.copy : string -> string
StringLabels.fill : bytes -> pos:int -> len:int -> char -> unit
StringLabels.uppercase : string -> string
StringLabels.lowercase : string -> string
StringLabels.capitalize : string -> string
StringLabels.uncapitalize : string -> string
StringLabels.get_uint8 : string -> int -> int
StringLabels.get_int8 : string -> int -> int
StringLabels.get_uint16_ne : string -> int -> int
StringLabels.get_uint16_be : string -> int -> int

Stdlib
StringLabels.get_uint16_le : string -> int -> int
StringLabels.get_int16_ne : string -> int -> int
StringLabels.get_int16_be : string -> int -> int
StringLabels.get_int16_le : string -> int -> int
StringLabels.get_int32_ne : string -> int -> int32
StringLabels.get_int32_be : string -> int -> int32
StringLabels.get_int32_le : string -> int -> int32
StringLabels.get_int64_ne : string -> int -> int64
StringLabels.get_int64_be : string -> int -> int64
StringLabels.get_int64_le : string -> int -> int64
String.blit : string -> int -> bytes -> int -> int -> unit
String.capitalize : string -> string
String.capitalize_ascii : string -> string
String.cat : string -> string -> string
String.compare : t -> t -> int
String.concat : string -> string list -> string
String.contains : string -> char -> bool
String.contains_from : string -> int -> char -> bool
String.copy : string -> string
String.empty : string
String.ends_with : suffix:string -> string -> bool
String.equal : t -> t -> bool
String.escaped : string -> string
String.exists : (char -> bool) -> string -> bool
String.fill : bytes -> int -> int -> char -> unit
String.fold_left : ('a -> char -> 'a) -> 'a -> string -> 'a
String.fold_right : (char -> 'a -> 'a) -> string -> 'a -> 'a
String.for_all : (char -> bool) -> string -> bool
String.get_int16_be : string -> int -> int
String.get_int16_le : string -> int -> int
String.get_int16_ne : string -> int -> int
String.get_int32_be : string -> int -> int32
String.get_int32_le : string -> int -> int32
String.get_int32_ne : string -> int -> int32

Stdlib
String.get_int64_be : string -> int -> int64
String.get_int64_le : string -> int -> int64
String.get_int64_ne : string -> int -> int64
String.get_int8 : string -> int -> int
String.get_uint16_be : string -> int -> int
String.get_uint16_le : string -> int -> int
String.get_uint16_ne : string -> int -> int
String.get_uint8 : string -> int -> int
String.get_utf_16be_uchar : t -> int -> Uchar.utf_decode
String.get_utf_16le_uchar : t -> int -> Uchar.utf_decode
String.get_utf_8_uchar : t -> int -> Uchar.utf_decode
String.index : string -> char -> int
String.index_from : string -> int -> char -> int
String.index_from_opt : string -> int -> char -> int option
String.index_opt : string -> char -> int option
String.init : int -> (int -> char) -> string
String.is_valid_utf_16be : t -> bool
String.is_valid_utf_16le : t -> bool
String.is_valid_utf_8 : t -> bool
String.iter : (char -> unit) -> string -> unit
String.iteri : (int -> char -> unit) -> string -> unit
String.lowercase : string -> string
String.lowercase_ascii : string -> string
String.make : int -> char -> string
String.map : (char -> char) -> string -> string
String.mapi : (int -> char -> char) -> string -> string
String.of_bytes : bytes -> string
String.of_seq : char Seq.t -> t
String.rcontains_from : string -> int -> char -> bool
String.rindex : string -> char -> int
String.rindex_from : string -> int -> char -> int
String.rindex_from_opt : string -> int -> char -> int option
String.rindex_opt : string -> char -> int option
String.split_on_char : char -> string -> string list

Stdlib
String.starts_with : prefix:string -> string -> bool
String.sub : string -> int -> int -> string
String.to_bytes : string -> bytes
String.to_seq : t -> char Seq.t
String.to_seqi : t -> (int * char) Seq.t
String.trim : string -> string
String.uncapitalize : string -> string
String.uncapitalize_ascii : string -> string
String.uppercase : string -> string
String.uppercase_ascii : string -> string