| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| | CCListLabels.( -- ) : int -> int -> int list | | |
| | CCListLabels.( --^ ) : int -> int -> int list | | |
| | CCListLabels.( <$> ) : ('a -> 'b) -> 'a list -> 'b list | | |
| | CCListLabels.( <*> ) : ('a -> 'b) list -> 'a list -> 'b list | | |
| | CCListLabels.( >>= ) : 'a list -> ('a -> 'b list) -> 'b list | | Base.List.( >>= ) : 'a list -> ('a -> 'b list) -> 'b list |
| | CCListLabels.( >\|= ) : 'a list -> ('a -> 'b) -> 'b list | | Base.List.( >>\| ) : 'a list -> ('a -> 'b) -> 'b list |
| | CCListLabels.( @ ) : 'a list -> 'a list -> 'a list | BatList.( @ ) : 'a list -> 'a list -> 'a list | |
| | CCListLabels.( and* ) : 'a list -> 'b list -> ('a * 'b) list | | |
| | CCListLabels.( and+ ) : 'a list -> 'b list -> ('a * 'b) list | | |
| | CCListLabels.( let* ) : 'a list -> ('a -> 'b list) -> 'b list | | |
| | CCListLabels.( let+ ) : 'a list -> ('a -> 'b) -> 'b list | | |
| | | | Base.List.Assoc.add : ('a, 'b) Base.List.Assoc.t -> equal:('a -> 'a -> bool) -> 'a -> 'b -> ('a, 'b) Base.List.Assoc.t |
| | | | Base.List.Assoc.find : ('a, 'b) Base.List.Assoc.t -> equal:('a -> 'a -> bool) -> 'a -> 'b option |
| | | | Base.List.Assoc.find_exn : ('a, 'b) Base.List.Assoc.t -> equal:('a -> 'a -> bool) -> 'a -> 'b |
| | CCListLabels.Assoc.get : eq:('a -> 'a -> bool) -> 'a -> ('a, 'b) CCListLabels.Assoc.t -> 'b option | | |
| | CCListLabels.Assoc.get_exn : eq:('a -> 'a -> bool) -> 'a -> ('a, 'b) CCListLabels.Assoc.t -> 'b | | |
| | | | Base.List.Assoc.inverse : ('a, 'b) Base.List.Assoc.t -> ('b, 'a) Base.List.Assoc.t |
| | | | Base.List.Assoc.map : ('a, 'b) Base.List.Assoc.t -> f:('b -> 'c) -> ('a, 'c) Base.List.Assoc.t |
| | CCListLabels.Assoc.mem : ?eq:('a -> 'a -> bool) -> 'a -> ('a, 'b) CCListLabels.Assoc.t -> bool | | Base.List.Assoc.mem : ('a, 'b) Base.List.Assoc.t -> equal:('a -> 'a -> bool) -> 'a -> bool |
| | CCListLabels.Assoc.remove : eq:('a -> 'a -> bool) -> 'a -> ('a, 'b) CCListLabels.Assoc.t -> ('a, 'b) CCListLabels.Assoc.t | | Base.List.Assoc.remove : ('a, 'b) Base.List.Assoc.t -> equal:('a -> 'a -> bool) -> 'a -> ('a, 'b) Base.List.Assoc.t |
| | CCListLabels.Assoc.set : eq:('a -> 'a -> bool) -> 'a -> 'b -> ('a, 'b) CCListLabels.Assoc.t -> ('a, 'b) CCListLabels.Assoc.t | | |
| | | | Base.List.Assoc.sexp_of_t : ('a -> Sexplib0.Sexp.t) -> ('b -> Sexplib0.Sexp.t) -> ('a, 'b) Base.List.Assoc.t -> Sexplib0.Sexp.t |
| | | | Base.List.Assoc.t_of_sexp : (Sexplib0.Sexp.t -> 'a) -> (Sexplib0.Sexp.t -> 'b) -> Sexplib0.Sexp.t -> ('a, 'b) Base.List.Assoc.t |
| | CCListLabels.Assoc.update : eq:('a -> 'a -> bool) -> f:('b option -> 'b option) -> 'a -> ('a, 'b) CCListLabels.Assoc.t -> ('a, 'b) CCListLabels.Assoc.t | | |
| | CCListLabels.Infix.( and& ) : 'a list -> 'b list -> ('a * 'b) list | | |
| ListLabels.[] : 'a list = ListLabels.[] | CCListLabels.[] : 'a list = [] | | |
| | CCListLabels.add_nodup : eq:('a -> 'a -> bool) -> 'a -> 'a list -> 'a list | | |
| | CCListLabels.all_ok : ('a, 'err) result list -> ('a CCListLabels.t, 'err) result | | |
| | CCListLabels.all_some : 'a option list -> 'a list option | | |
| | | | Base.List.all_unit : unit list list -> unit list |
| ListLabels.append : 'a list -> 'a list -> 'a list | CCListLabels.append : 'a list -> 'a list -> 'a list | | Base.List.append : 'a list -> 'a list -> 'a list |
| ListLabels.assoc : 'a -> ('a * 'b) list -> 'b | CCListLabels.assoc : eq:('a -> 'a -> bool) -> 'a -> ('a * 'b) list -> 'b | | |
| ListLabels.assoc_opt : 'a -> ('a * 'b) list -> 'b option | CCListLabels.assoc_opt : eq:('a -> 'a -> bool) -> 'a -> ('a * 'b) list -> 'b option | | |
| ListLabels.assq : 'a -> ('a * 'b) list -> 'b | CCListLabels.assq : 'a -> ('a * 'b) list -> 'b | | |
| ListLabels.assq_opt : 'a -> ('a * 'b) list -> 'b | CCListLabels.assq_opt : 'a -> ('a * 'b) list -> 'b option | | |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| option | | | |
| | | | Base.List.bind : 'a list -> f:('a -> 'b list) -> 'b list |
| | CCListLabels.cartesian_product : 'a list list -> 'a list list | BatList.n_cartesian_product : 'a list list -> 'a list list | Base.List.all : 'a list list -> 'a list list |
| | | BatList.cartesian_product : 'a list -> 'b list -> ('a * 'b) list | Base.List.cartesian_product : 'a list -> 'b list -> ('a * 'b) list |
| | CCListLabels.chunks : int -> 'a list -> 'a list list | | Base.List.chunks_of : 'a list -> length:int -> 'a list list |
| ListLabels.combine : 'a list -> 'b list -> ('a * 'b) list | CCListLabels.combine : 'a list -> 'b list -> ('a * 'b) list | BatList.combine : 'a list -> 'b list -> ('a * 'b) list | Base.List.zip_exn : 'a list -> 'b list -> ('a * 'b) list |
| | CCListLabels.combine_gen : 'a list -> 'b list -> ('a * 'b) CCListLabels.gen | | |
| | CCListLabels.combine_shortest : 'a list -> 'b list -> ('a * 'b) list | | |
| ListLabels.compare : cmp:('a -> 'a -> int) -> 'a list -> 'a list -> int | CCListLabels.compare : ('a -> 'a -> int) -> 'a list -> 'a list -> int | | Base.List.compare : ('a -> 'a -> int) -> 'a list -> 'a list -> int |
| ListLabels.compare_length_with : 'a list -> len:int -> int | CCListLabels.compare_length_with : 'a list -> int -> int | | |
| ListLabels.compare_lengths : 'a list -> 'b list -> int | CCListLabels.compare_lengths : 'a list -> 'b list -> int | | |
| ListLabels.concat : 'a list list -> 'a list | CCListLabels.concat : 'a list list -> 'a list | | Base.List.concat : 'a list list -> 'a list |
| ListLabels.concat_map : f:('a -> 'b list) -> 'a list -> 'b list | CCListLabels.concat_map : f:('a -> 'b list) -> 'a list -> 'b list | BatList.Labels.concat_map : f:('a -> 'b list) -> 'a list -> 'b list | Base.List.concat_map : 'a list -> f:('a -> 'b list) -> 'b list |
| | | | Base.List.concat_mapi : 'a list -> f:(int -> 'a -> 'b list) -> 'b list |
| | | | Base.List.concat_no_order : 'a list list -> 'a list |
| ListLabels.cons : 'a -> 'a list -> 'a list | CCListLabels.cons : 'a -> 'a list -> 'a list | BatList.cons : 'a -> 'a list -> 'a list | Base.List.cons : 'a -> 'a list -> 'a list |
| | CCListLabels.cons' : 'a list -> 'a -> 'a list | | |
| | CCListLabels.cons_maybe : 'a option -> 'a list -> 'a list | | |
| | | | Base.List.contains_dup : compare:('a -> 'a -> int) -> 'a list -> bool |
| | CCListLabels.count : f:('a -> bool) -> 'a list -> int | BatList.Labels.count_matching : f:('a -> bool) -> 'a list -> int | Base.List.count : 'a list -> f:('a -> bool) -> int |
| | | | Base.List.counti : 'a list -> f:(int -> 'a -> bool) -> int |
| | CCListLabels.count_true_false : f:('a -> bool) -> 'a list -> int * int | | |
| | | | Base.List.dedup_and_sort : compare:('a -> 'a -> int) -> 'a list -> 'a list |
| | CCListLabels.diagonal : 'a list -> ('a * 'a) list | | |
| | CCListLabels.drop : int -> 'a list -> 'a list | BatList.drop : int -> 'a list -> 'a list | Base.List.drop : 'a list -> int -> 'a list |
| | | | Base.List.drop_last : 'a list -> 'a list option |
| | | | Base.List.drop_last_exn : 'a list -> 'a list |
| | CCListLabels.drop_while : f:('a -> bool) -> 'a list -> 'a list | BatList.Labels.drop_while : f:('a -> bool) -> 'a list -> 'a list | Base.List.drop_while : 'a list -> f:('a -> bool) -> 'a list |
| | CCListLabels.empty : 'a list = [] | | |
| ListLabels.equal : eq:('a -> 'a -> bool) -> 'a list -> 'a list -> bool | CCListLabels.equal : ('a -> 'a -> bool) -> 'a list -> 'a list -> bool | BatList.equal : ('a -> 'a -> bool) -> 'a list -> 'a list -> bool | Base.List.equal : ('a -> 'a -> bool) -> 'a list -> 'a list -> bool |
| ListLabels.exists : f:('a -> bool) -> 'a list -> bool | CCListLabels.exists : f:('a -> bool) -> 'a list -> bool | BatList.Labels.exists : f:('a -> bool) -> 'a list -> bool | Base.List.exists : 'a list -> f:('a -> bool) -> bool |
| ListLabels.exists2 : f:('a -> 'b -> bool) -> 'a list -> 'b list -> bool | CCListLabels.exists2 : f:('a -> 'b -> bool) -> 'a list -> 'b list -> bool | BatList.Labels.exists2 : f:('a -> 'b -> bool) -> 'a list -> 'b list -> bool | Base.List.exists2_exn : 'a list -> 'b list -> f:('a -> 'b -> bool) -> bool |
| | | | Base.List.exists2 : 'a list -> 'b list -> f:('a -> 'b -> bool) -> bool Base.List.Or_unequal_lengths.t |
| | | | Base.List.existsi : 'a list -> f:(int -> 'a -> bool) -> bool |
| ListLabels.fast_sort : cmp:('a -> 'a -> int) -> 'a list -> 'a list | CCListLabels.fast_sort : cmp:('a -> 'a -> int) -> 'a list -> 'a list | BatList.Labels.fast_sort : ?cmp:('a -> 'a -> int) -> 'a list -> 'a list | |
| ListLabels.filter : f:('a -> bool) -> 'a list -> 'a | CCListLabels.filter : f:('a -> bool) -> 'a list -> 'a list | BatList.Labels.filter : f:('a -> bool) -> 'a list -> 'a list | Base.List.filter : 'a list -> f:('a -> bool) -> 'a list |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| list | | | |
| ListLabels.filter_map : f:('a -> 'b option) -> 'a list -> 'b list | CCListLabels.filter_map : f:('a -> 'b option) -> 'a list -> 'b list | BatList.Labels.filter_map : f:('a -> 'b option) -> 'a list -> 'b list | Base.List.filter_map : 'a list -> f:('a -> 'b option) -> 'b list |
| | | | Base.List.filter_mapi : 'a list -> f:(int -> 'a -> 'b option) -> 'b list |
| | | | Base.List.filter_opt : 'a option list -> 'a list |
| ListLabels.filteri : f:(int -> 'a -> bool) -> 'a list -> 'a list | CCListLabels.filteri : f:(int -> 'a -> bool) -> 'a list -> 'a list | | Base.List.filteri : 'a list -> f:(int -> 'a -> bool) -> 'a list |
| ListLabels.find : f:('a -> bool) -> 'a list -> 'a | CCListLabels.find : f:('a -> bool) -> 'a list -> 'a | BatList.Labels.find : f:('a -> bool) -> 'a list -> 'a | Base.List.find_exn : 'a list -> f:('a -> bool) -> 'a |
| ListLabels.find_all : f:('a -> bool) -> 'a list -> 'a list | CCListLabels.find_all : f:('a -> bool) -> 'a list -> 'a list | BatList.Labels.find_all : f:('a -> bool) -> 'a list -> 'a list | |
| | | | Base.List.find_a_dup : compare:('a -> 'a -> int) -> 'a list -> 'a option |
| | | | Base.List.find_all_dups : compare:('a -> 'a -> int) -> 'a list -> 'a list |
| | | | Base.List.find_consecutive_duplicate : 'a list -> equal:('a -> 'a -> bool) -> ('a * 'a) option |
| | | BatList.Labels.find_exn : f:('a -> bool) -> exn -> 'a list -> 'a | |
| | CCListLabels.find_idx : f:('a -> bool) -> 'a list -> (int * 'a) option | | |
| ListLabels.find_map : f:('a -> 'b option) -> 'a list -> 'b option | CCListLabels.find_map : f:('a -> 'b option) -> 'a list -> 'b option | BatList.Labels.find_map_opt : f:('a -> 'b option) -> 'a list -> 'b option | Base.List.find_map : 'a list -> f:('a -> 'b option) -> 'b option |
| | | | Base.List.find_map_exn : 'a list -> f:('a -> 'b option) -> 'b |
| | CCListLabels.find_mapi : f:(int -> 'a -> 'b option) -> 'a list -> 'b option | | Base.List.find_mapi : 'a list -> f:(int -> 'a -> 'b option) -> 'b option |
| | | | Base.List.find_mapi_exn : 'a list -> f:(int -> 'a -> 'b option) -> 'b |
| ListLabels.find_opt : f:('a -> bool) -> 'a list -> 'a option | CCListLabels.find_opt : f:('a -> bool) -> 'a list -> 'a option | | Base.List.find : 'a list -> f:('a -> bool) -> 'a option |
| | CCListLabels.find_pred : f:('a -> bool) -> 'a list -> 'a option | | |
| | CCListLabels.find_pred_exn : f:('a -> bool) -> 'a list -> 'a | | |
| | | BatList.Labels.findi : f:(int -> 'a -> bool) -> 'a list -> int * 'a | |
| | | | Base.List.findi : 'a list -> f:(int -> 'a -> bool) -> (int * 'a) option |
| | CCListLabels.flat_map : f:('a -> 'b list) -> 'a list -> 'b list | | |
| | CCListLabels.flat_map_i : f:(int -> 'a -> 'b list) -> 'a list -> 'b list | | |
| ListLabels.flatten : 'a list list -> 'a list | CCListLabels.flatten : 'a list list -> 'a list | | |
| | | BatList.Labels.fold : f:('a -> 'b -> 'a) -> init:'a -> 'b list -> 'a | Base.List.fold : 'a list -> init:'accum -> f:('accum -> 'a -> 'accum) -> 'accum |
| | | | Base.List.fold2 : 'a list -> 'b list -> init:'c -> f:('c -> 'a -> 'b -> 'c) -> 'c Base.List.Or_unequal_lengths.t |
| | | | Base.List.fold2_exn : 'a list -> 'b list -> init:'c -> f:('c -> 'a -> 'b -> 'c) -> 'c |
| | CCListLabels.fold_filter_map : f:('acc -> 'a -> 'acc * 'b option) -> init:'acc -> 'a list -> 'acc * 'b list | | |
| | CCListLabels.fold_filter_map_i : f:('acc -> int -> 'a -> 'acc * 'b option) -> init:'acc -> 'a list -> 'acc * 'b list | | |
| | CCListLabels.fold_flat_map : f:('acc -> 'a -> 'acc * 'b list) -> init:'acc -> 'a list -> 'acc * 'b list | | |
| | CCListLabels.fold_flat_map_i : f:('acc -> int -> 'a -> 'acc * 'b list) -> init:'acc -> 'a list -> 'acc * 'b list | | |
| ListLabels.fold_left : f:('a -> 'b -> 'a) -> init:'a -> 'b list -> 'a | CCListLabels.fold_left : f:('a -> 'b -> 'a) -> init:'a -> 'b list -> 'a | BatList.Labels.fold_left : f:('a -> 'b -> 'a) -> init:'a -> 'b list -> 'a | Base.List.fold_left : 'a list -> init:'b -> f:('b -> 'a -> 'b) -> 'b |
| ListLabels.fold_left2 : f:('a -> 'b -> 'c -> 'a) -> init:'a -> 'b list -> 'c list -> 'a | CCListLabels.fold_left2 : f:('a -> 'b -> 'c -> 'a) -> init:'a -> 'b list -> 'c list -> 'a | BatList.Labels.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 | CCListLabels.fold_left_map : f:('a -> 'b -> 'a * 'c) -> init:'a -> 'b list -> 'a * 'c list | | |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| | CCListLabels.fold_map : f:('acc -> 'a -> 'acc * 'b) -> init:'acc -> 'a list -> 'acc * 'b list | | Base.List.fold_map : 'a list -> init:'b -> f:('b -> 'a -> 'b * 'c) -> 'b * 'c list |
| | CCListLabels.fold_map2 : f:('acc -> 'a -> 'b -> 'acc * 'c) -> init:'acc -> 'a list -> 'b list -> 'acc * 'c list | | |
| | CCListLabels.fold_map_i : f:('acc -> int -> 'a -> 'acc * 'b) -> init:'acc -> 'a list -> 'acc * 'b list | | Base.List.fold_mapi : 'a list -> init:'b -> f:(int -> 'b -> 'a -> 'b * 'c) -> 'b * 'c list |
| | CCListLabels.fold_on_map : f:('a -> 'b) -> reduce:('acc -> 'b -> 'acc) -> init:'acc -> 'a list -> 'acc | | |
| | CCListLabels.fold_product : f:('c -> 'a -> 'b -> 'c) -> init:'c -> 'a list -> 'b list -> 'c | | |
| | | | Base.List.fold_result : 'a list -> init:'accum -> f:('accum -> 'a -> ('accum, 'e) Base.Result.t) -> ('accum, 'e) Base.Result.t |
| ListLabels.fold_right : f:('a -> 'b -> 'b) -> 'a list -> init:'b -> 'b | CCListLabels.fold_right : f:('a -> 'b -> 'b) -> 'a list -> init:'b -> 'b | BatList.Labels.fold_right : f:('a -> 'b -> 'b) -> 'a list -> init:'b -> 'b | Base.List.fold_right : 'a list -> f:('a -> 'b -> 'b) -> init:'b -> 'b |
| ListLabels.fold_right2 : f:('a -> 'b -> 'c -> 'c) -> 'a list -> 'b list -> init:'c -> 'c | CCListLabels.fold_right2 : f:('a -> 'b -> 'c -> 'c) -> 'a list -> 'b list -> init:'c -> 'c | BatList.Labels.fold_right2 : f:('a -> 'b -> 'c -> 'c) -> 'a list -> 'b list -> init:'c -> 'c | |
| | | | Base.List.fold_until : 'a list -> init:'accum -> f:('accum -> 'a -> ('accum, 'final) Base.Container_intf.Continue_or_stop.t) -> finish:('accum -> 'final) -> 'final |
| | CCListLabels.fold_while : f:('a -> 'b -> 'a * [ `Continue \| `Stop ]) -> init:'a -> 'b list -> 'a | | |
| | CCListLabels.foldi : f:('b -> int -> 'a -> 'b) -> init:'b -> 'a list -> 'b | | Base.List.foldi : 'a list -> init:'b -> f:(int -> 'b -> 'a -> 'b) -> 'b |
| | CCListLabels.foldi2 : f:('c -> int -> 'a -> 'b -> 'c) -> init:'c -> 'a list -> 'b list -> 'c | | |
| | | | Base.List.folding_map : 'a list -> init:'b -> f:('b -> 'a -> 'b * 'c) -> 'c list |
| | | | Base.List.folding_mapi : 'a list -> init:'b -> f:(int -> 'b -> 'a -> 'b * 'c) -> 'c list |
| ListLabels.for_all : f:('a -> bool) -> 'a list -> bool | CCListLabels.for_all : f:('a -> bool) -> 'a list -> bool | BatList.Labels.for_all : f:('a -> bool) -> 'a list -> bool | Base.List.for_all : 'a list -> f:('a -> bool) -> bool |
| ListLabels.for_all2 : f:('a -> 'b -> bool) -> 'a list -> 'b list -> bool | CCListLabels.for_all2 : f:('a -> 'b -> bool) -> 'a list -> 'b list -> bool | BatList.Labels.for_all2 : f:('a -> 'b -> bool) -> 'a list -> 'b list -> bool | Base.List.for_all2_exn : 'a list -> 'b list -> f:('a -> 'b -> bool) -> bool |
| | | | Base.List.for_all2 : 'a list -> 'b list -> f:('a -> 'b -> bool) -> bool Base.List.Or_unequal_lengths.t |
| | | | Base.List.for_alli : 'a list -> f:(int -> 'a -> bool) -> bool |
| | CCListLabels.get_at_idx : int -> 'a list -> 'a option | | |
| | CCListLabels.get_at_idx_exn : int -> 'a list -> 'a | | |
| | | | Base.List.groupi : 'a list -> break:(int -> 'a -> 'a -> bool) -> 'a list list |
| | CCListLabels.group_by : ?hash:('a -> int) -> ?eq:('a -> 'a -> bool) -> 'a list -> 'a list list | | Base.List.group : 'a list -> break:('a -> 'a -> bool) -> 'a list list |
| | CCListLabels.group_join_by : ?eq:('a -> 'a -> bool) -> ?hash:('a -> int) -> ('b -> 'a) -> 'a list -> 'b list -> ('a * 'b list) list | | |
| | CCListLabels.group_succ : eq:('a -> 'a -> bool) -> 'a list -> 'a list list | | |
| | | | Base.List.hash_fold_t : (Base.Ppx_hash_lib.Std.Hash.state -> 'a -> Base.Ppx_hash_lib.Std.Hash.state) -> Base.Ppx_hash_lib.Std.Hash.state -> 'a list -> Base.Ppx_hash_lib.Std.Hash.state |
| ListLabels.hd : 'a list -> 'a | CCListLabels.hd : 'a list -> 'a | BatList.hd : 'a list -> 'a | Base.List.hd_exn : 'a list -> 'a |
| | CCListLabels.hd_tl : 'a list -> 'a * 'a list | | |
| | CCListLabels.head_opt : 'a list -> 'a option | | Base.List.hd : 'a list -> 'a option |
| | | | Base.List.ignore_m : 'a list -> unit list |
| ListLabels.init : len:int -> f:(int -> 'a) -> 'a list | CCListLabels.init : int -> f:(int -> 'a) -> 'a list | BatList.Labels.init : int -> f:(int -> 'a) -> 'a list | Base.List.init : int -> f:(int -> 'a) -> 'a list |
| | CCListLabels.insert_at_idx : int -> 'a -> 'a list -> 'a list | | |
| | CCListLabels.inter : eq:('a -> 'a -> bool) -> 'a list -> 'a list -> 'a list | | |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| | CCListLabels.interleave : 'a list -> 'a list -> 'a list | | |
| | CCListLabels.intersperse : x:'a -> 'a list -> 'a list | | Base.List.intersperse : 'a list -> sep:'a -> 'a list |
| | | | Base.List.invariant : 'a Base.Invariant_intf.inv -> 'a list Base.Invariant_intf.inv |
| | CCListLabels.is_empty : 'a list -> bool | | Base.List.is_empty : 'a list -> bool |
| | | | Base.List.is_prefix : 'a list -> prefix:'a list -> equal:('a -> 'a -> bool) -> bool |
| | CCListLabels.is_sorted : cmp:('a -> 'a -> int) -> 'a list -> bool | | Base.List.is_sorted : 'a list -> compare:('a -> 'a -> int) -> bool |
| | | | Base.List.is_sorted_strictly : 'a list -> compare:('a -> 'a -> int) -> bool |
| | | | Base.List.is_suffix : 'a list -> suffix:'a list -> equal:('a -> 'a -> bool) -> bool |
| ListLabels.iter : f:('a -> unit) -> 'a list -> unit | CCListLabels.iter : f:('a -> unit) -> 'a list -> unit | BatList.Labels.iter : f:('a -> unit) -> 'a list -> unit | Base.List.iter : 'a list -> f:('a -> unit) -> unit |
| ListLabels.iter2 : f:('a -> 'b -> unit) -> 'a list -> 'b list -> unit | CCListLabels.iter2 : f:('a -> 'b -> unit) -> 'a list -> 'b list -> unit | BatList.Labels.iter2 : f:('a -> 'b -> unit) -> 'a list -> 'b list -> unit | Base.List.iter2_exn : 'a list -> 'b list -> f:('a -> 'b -> unit) -> unit |
| | | | Base.List.iter2 : 'a list -> 'b list -> f:('a -> 'b -> unit) -> unit Base.List.Or_unequal_lengths.t |
| ListLabels.iteri : f:(int -> 'a -> unit) -> 'a list -> unit | CCListLabels.iteri : f:(int -> 'a -> unit) -> 'a list -> unit | BatList.Labels.iteri : f:(int -> 'a -> unit) -> 'a list -> unit | Base.List.iteri : 'a list -> f:(int -> 'a -> unit) -> unit |
| | CCListLabels.iteri2 : f:(int -> 'a -> 'b -> unit) -> 'a list -> 'b list -> unit | | |
| | | | Base.List.join : 'a list list -> 'a list |
| | CCListLabels.join : join_row:('a -> 'b -> 'c option) -> 'a list -> 'b list -> 'c list | | |
| | CCListLabels.join_all_by : ?eq:('key -> 'key -> bool) -> ?hash:('key -> int) -> ('a -> 'key) -> ('b -> 'key) -> merge:('key -> 'a list -> 'b list -> 'c option) -> 'a list -> 'b list -> 'c list | | |
| | CCListLabels.join_by : ?eq:('key -> 'key -> bool) -> ?hash:('key -> int) -> ('a -> 'key) -> ('b -> 'key) -> merge:('key -> 'a -> 'b -> 'c option) -> 'a list -> 'b list -> 'c list | | |
| | CCListLabels.keep_ok : ('a, 'b) result list -> 'a list | | |
| | CCListLabels.keep_some : 'a option list -> 'a list | | |
| | CCListLabels.last : int -> 'a list -> 'a list | | |
| | | BatList.last : 'a list -> 'a | Base.List.last_exn : 'a list -> 'a |
| | CCListLabels.last_opt : 'a list -> 'a option | | Base.List.last : 'a list -> 'a option |
| ListLabels.length : 'a list -> int | CCListLabels.length : 'a list -> int | | Base.List.length : 'a list -> int |
| ListLabels.map : f:('a -> 'b) -> 'a list -> 'b list | CCListLabels.map : f:('a -> 'b) -> 'a list -> 'b list | BatList.Labels.map : f:('a -> 'b) -> 'a list -> 'b list | Base.List.map : 'a list -> f:('a -> 'b) -> 'b list |
| ListLabels.map2 : f:('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | CCListLabels.map2 : f:('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | BatList.Labels.map2 : f:('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | Base.List.map2_exn : 'a list -> 'b list -> f:('a -> 'b -> 'c) -> 'c list |
| | | | Base.List.map2 : 'a list -> 'b list -> f:('a -> 'b -> 'c) -> 'c list Base.List.Or_unequal_lengths.t |
| | | | Base.List.map3 : 'a list -> 'b list -> 'c list -> f:('a -> 'b -> 'c -> 'd) -> 'd list Base.List.Or_unequal_lengths.t |
| | | | Base.List.map3_exn : 'a list -> 'b list -> 'c list -> f:('a -> 'b -> 'c -> 'd) -> 'd list |
| | CCListLabels.map_product_l : f:('a -> 'b list) -> 'a list -> 'b list list | | |
| ListLabels.mapi : f:(int -> 'a -> 'b) -> 'a list -> 'b list | CCListLabels.mapi : f:(int -> 'a -> 'b) -> 'a list -> 'b list | BatList.Labels.mapi : f:(int -> 'a -> 'b) -> 'a list -> 'b list | Base.List.mapi : 'a list -> f:(int -> 'a -> 'b) -> 'b list |
| | | | Base.List.max_elt : 'a list -> compare:('a -> 'a -> int) -> 'a option |
| ListLabels.mem : 'a -> set:'a list -> bool | CCListLabels.mem : ?eq:('a -> 'a -> bool) -> 'a -> 'a list -> bool | | Base.List.mem : 'a list -> 'a -> equal:('a -> 'a -> bool) -> bool |
| ListLabels.mem_assoc : 'a -> map:('a * 'b) list -> bool | CCListLabels.mem_assoc : ?eq:('a -> 'a -> bool) -> 'a -> ('a * 'b) list -> bool | | |
| ListLabels.mem_assq : 'a -> map:('a * 'b) list -> bool | CCListLabels.mem_assq : 'a -> map:('a * 'b) list -> bool | | |
| ListLabels.memq : 'a -> set:'a list -> bool | CCListLabels.memq : 'a -> set:'a list -> bool | | |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| ListLabels.merge : cmp:('a -> 'a -> int) -> 'a list -> 'a list -> 'a list | CCListLabels.merge : cmp:('a -> 'a -> int) -> 'a list -> 'a list -> 'a list | BatList.Labels.merge : ?cmp:('a -> 'a -> int) -> 'a list -> 'a list -> 'a list | Base.List.merge : 'a list -> 'a list -> compare:('a -> 'a -> int) -> 'a list |
| | CCListLabels.mguard : bool -> unit list | | |
| | | | Base.List.min_elt : 'a list -> compare:('a -> 'a -> int) -> 'a option |
| ListLabels.nth : 'a list -> int -> 'a | CCListLabels.nth : 'a list -> int -> 'a | | Base.List.nth_exn : 'a list -> int -> 'a |
| ListLabels.nth_opt : 'a list -> int -> 'a option | CCListLabels.nth_opt : 'a list -> int -> 'a option | | Base.List.nth : 'a list -> int -> 'a option |
| | CCListLabels.of_gen : 'a CCListLabels.gen -> 'a list | | |
| | CCListLabels.of_iter : 'a CCListLabels.iter -> 'a list | | |
| | | | Base.List.of_list : 'a list -> 'a list |
| ListLabels.of_seq : 'a Seq.t -> 'a list | CCListLabels.of_seq : 'a Seq.t -> 'a list | | |
| | CCListLabels.of_seq_rev : 'a Seq.t -> 'a list | | |
| ListLabels.partition : f:('a -> bool) -> 'a list -> 'a list * 'a list | CCListLabels.partition : f:('a -> bool) -> 'a list -> 'a list * 'a list | BatList.Labels.partition : f:('a -> bool) -> 'a list -> 'a list * 'a list | Base.List.partition_tf : 'a list -> f:('a -> bool) -> 'a list * 'a list |
| | | | Base.List.partition3_map : 'a list -> f:('a -> [ `Fst of 'b \| `Snd of 'c \| `Trd of 'd ]) -> 'b list * 'c list * 'd list |
| | CCListLabels.partition_filter_map : f:('a -> [< `Drop \| `Left of 'b \| `Right of 'c ]) -> 'a list -> 'b list * 'c list | | |
| | CCListLabels.partition_map : f:('a -> [< `Drop \| `Left of 'b \| `Right of 'c ]) -> 'a list -> 'b list * 'c list | | |
| ListLabels.partition_map : f:('a -> ('b, 'c) Either.t) -> 'a list -> 'b list * 'c list | CCListLabels.partition_map_either : f:('a -> ('b, 'c) CCEither.t) -> 'a list -> 'b list * 'c list | BatList.Labels.partition_map : f:('a -> ('b, 'c) BatEither.t) -> 'a list -> 'b list * 'c list | Base.List.partition_map : 'a list -> f:('a -> ('b, 'c) Base.Either0.t) -> 'b list * 'c list |
| | | | Base.List.partition_result : ('ok, 'error) Base.Result.t list -> 'ok list * 'error list |
| | | | Base.List.permute : ?random_state:Base.Random.State.t -> 'a list -> 'a list |
| | CCListLabels.pp : ?pp_start:unit CCListLabels.printer -> ?pp_stop:unit CCListLabels.printer -> ?pp_sep:unit CCListLabels.printer -> 'a CCListLabels.printer -> 'a list CCListLabels.printer | | |
| | CCListLabels.product : f:('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | | |
| | CCListLabels.pure : 'a -> 'a list | | |
| | CCListLabels.random : 'a CCListLabels.random_gen -> 'a list CCListLabels.random_gen | | |
| | CCListLabels.random_choose : 'a list -> 'a CCListLabels.random_gen | | |
| | | | Base.List.random_element : ?random_state:Base.Random.State.t -> 'a list -> 'a option |
| | | | Base.List.random_element_exn : ?random_state:Base.Random.State.t -> 'a list -> 'a |
| | CCListLabels.random_len : int -> 'a CCListLabels.random_gen -> 'a list CCListLabels.random_gen | | |
| | CCListLabels.random_non_empty : 'a CCListLabels.random_gen -> 'a list CCListLabels.random_gen | | |
| | CCListLabels.random_sequence : 'a CCListLabels.random_gen list -> 'a list CCListLabels.random_gen | | |
| | CCListLabels.range : int -> int -> int list | | Base.List.range : ?stride:int -> ?start:[ `exclusive \| `inclusive ] -> ?stop:[ `exclusive \| `inclusive ] -> int -> int -> int list |
| | CCListLabels.range' : int -> int -> int list | | |
| | | | Base.List.range' : compare:('a -> 'a -> int) -> stride:('a -> 'a) -> ?start:[ `exclusive \| `inclusive ] -> ?stop:[ `exclusive \| `inclusive ] -> 'a -> 'a -> 'a list |
| | CCListLabels.range_by : step:int -> int -> int -> int list | | |
| | CCListLabels.reduce : f:('a -> 'a -> 'a) -> 'a list -> 'a option | | Base.List.reduce : 'a list -> f:('a -> 'a -> 'a) -> 'a option |
| | | | Base.List.reduce_balanced : 'a list -> f:('a -> 'a -> 'a) -> 'a option |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| | | | Base.List.reduce_balanced_exn : 'a list -> f:('a -> 'a -> 'a) -> 'a |
| | CCListLabels.reduce_exn : f:('a -> 'a -> 'a) -> 'a list -> 'a | | Base.List.reduce_exn : 'a list -> f:('a -> 'a -> 'a) -> 'a |
| | CCListLabels.remove : eq:('a -> 'a -> bool) -> key:'a -> 'a list -> 'a list | | |
| ListLabels.remove_assoc : 'a -> ('a * 'b) list -> ('a * 'b) list | CCListLabels.remove_assoc : eq:('a -> 'a -> bool) -> 'a -> ('a * 'b) list -> ('a * 'b) list | | |
| ListLabels.remove_assq : 'a -> ('a * 'b) list -> ('a * 'b) list | CCListLabels.remove_assq : 'a -> ('a * 'b) list -> ('a * 'b) list | | |
| | CCListLabels.remove_at_idx : int -> 'a list -> 'a list | | |
| | | | Base.List.remove_consecutive_duplicates : ?which_to_keep:[ `First \| `Last ] -> 'a list -> equal:('a -> 'a -> bool) -> 'a list |
| | | BatList.Labels.remove_if : f:('a -> bool) -> 'a list -> 'a list | |
| | CCListLabels.remove_one : eq:('a -> 'a -> bool) -> 'a -> 'a list -> 'a list | | |
| | CCListLabels.repeat : int -> 'a list -> 'a list | | |
| | CCListLabels.replicate : int -> 'a -> 'a list | | |
| | CCListLabels.return : 'a -> 'a list | | Base.List.return : 'a -> 'a list |
| ListLabels.rev : 'a list -> 'a list | CCListLabels.rev : 'a list -> 'a list | BatList.rev : 'a list -> 'a list | Base.List.rev : 'a list -> 'a list |
| ListLabels.rev_append : 'a list -> 'a list -> 'a list | CCListLabels.rev_append : 'a list -> 'a list -> 'a list | BatList.rev_append : 'a list -> 'a list -> 'a list | Base.List.rev_append : 'a list -> 'a list -> 'a list |
| | | | Base.List.rev_filter : 'a list -> f:('a -> bool) -> 'a list |
| | | | Base.List.rev_filter_map : 'a list -> f:('a -> 'b option) -> 'b list |
| | | | Base.List.rev_filter_mapi : 'a list -> f:(int -> 'a -> 'b option) -> 'b list |
| ListLabels.rev_map : f:('a -> 'b) -> 'a list -> 'b list | CCListLabels.rev_map : f:('a -> 'b) -> 'a list -> 'b list | BatList.Labels.rev_map : f:('a -> 'b) -> 'a list -> 'b list | Base.List.rev_map : 'a list -> f:('a -> 'b) -> 'b list |
| ListLabels.rev_map2 : f:('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | CCListLabels.rev_map2 : f:('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | BatList.Labels.rev_map2 : f:('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | Base.List.rev_map2_exn : 'a list -> 'b list -> f:('a -> 'b -> 'c) -> 'c list |
| | | | Base.List.rev_map2 : 'a list -> 'b list -> f:('a -> 'b -> 'c) -> 'c list Base.List.Or_unequal_lengths.t |
| | | | Base.List.rev_map3 : 'a list -> 'b list -> 'c list -> f:('a -> 'b -> 'c -> 'd) -> 'd list Base.List.Or_unequal_lengths.t |
| | | | Base.List.rev_map3_exn : 'a list -> 'b list -> 'c list -> f:('a -> 'b -> 'c -> 'd) -> 'd list |
| | | | Base.List.rev_map_append : 'a list -> 'b list -> f:('a -> 'b) -> 'b list |
| | | | Base.List.rev_mapi : 'a list -> f:(int -> 'a -> 'b) -> 'b list |
| | | BatList.Labels.rfind : f:('a -> bool) -> 'a list -> 'a | |
| | CCListLabels.scan_left : f:('acc -> 'a -> 'acc) -> init:'acc -> 'a list -> 'acc list | | |
| | CCListLabels.set_at_idx : int -> 'a -> 'a list -> 'a list | | |
| | | | Base.List.sexp_of_t : ('a -> Sexplib0.Sexp.t) -> 'a list -> Sexplib0.Sexp.t |
| ListLabels.sort : cmp:('a -> 'a -> int) -> 'a list -> 'a list | CCListLabels.sort : cmp:('a -> 'a -> int) -> 'a list -> 'a list | | Base.List.sort : 'a list -> compare:('a -> 'a -> int) -> 'a list |
| ListLabels.sort_uniq : cmp:('a -> 'a -> int) -> 'a list -> 'a list | CCListLabels.sort_uniq : cmp:('a -> 'a -> int) -> 'a list -> 'a list | | |
| | CCListLabels.sorted_insert : cmp:('a -> 'a -> int) -> ?uniq:bool -> 'a -> 'a list -> 'a list | | |
| | CCListLabels.sorted_merge : cmp:('a -> 'a -> int) -> 'a list -> 'a list -> 'a list | | |
| | CCListLabels.sorted_merge_uniq : cmp:('a -> 'a -> int) -> 'a list -> 'a list -> 'a list | | |
| ListLabels.split : ('a * 'b) list -> 'a list * 'b list | CCListLabels.split : ('a * 'b) list -> 'a list * 'b list | | Base |
| | | | Base.List.split_n : 'a list -> int -> 'a list * 'a list |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| | | | Base.List.split_while : 'a list -> f:('a -> bool) -> 'a list * 'a list |
| ListLabels.stable_sort : cmp:('a -> 'a -> int) -> 'a list -> 'a list | CCListLabels.stable_sort : cmp:('a -> 'a -> int) -> 'a list -> 'a list | BatList.Labels.stable_sort : ?cmp:('a -> 'a -> int) -> 'a list -> 'a list | Base.List.stable_sort : 'a list -> compare:('a -> 'a -> int) -> 'a list |
| | | | Base.List.sub : 'a list -> pos:int -> len:int -> 'a list |
| | CCListLabels.sublists_of_len : ?last:('a list -> 'a list option) -> ?offset:int -> len:int -> 'a list -> 'a list list | | |
| | CCListLabels.subset : eq:('a -> 'a -> bool) -> 'a list -> 'a list -> bool | | |
| | | BatList.Labels.subset : cmp:('a -> 'b -> int) -> 'a list -> 'b list -> bool | |
| | | | Base.List.sum : (module Base.Container_intf.Summable with type t = 'sum) -> 'a list -> f:('a -> 'sum) -> 'sum |
| | | | Base.List.t_of_sexp : (Sexplib0.Sexp.t -> 'a) -> Sexplib0.Sexp.t -> 'a list |
| | | | Base.List.t_sexp_grammar : Base.Ppx_sexp_conv_lib.Sexp.Private.Raw_grammar.t... |
| | CCListLabels.tail_opt : 'a list -> 'a list option | | |
| | CCListLabels.take : int -> 'a list -> 'a list | BatList.take : int -> 'a list -> 'a list | Base.List.take : 'a list -> int -> 'a list |
| | CCListLabels.take_drop : int -> 'a list -> 'a list * 'a list | | |
| | CCListLabels.take_drop_while : f:('a -> bool) -> 'a list -> 'a list * 'a list | | |
| | CCListLabels.take_while : f:('a -> bool) -> 'a list -> 'a list | BatList.Labels.take_while : f:('a -> bool) -> 'a list -> 'a list | Base.List.take_while : 'a list -> f:('a -> bool) -> 'a list |
| ListLabels.tl : 'a list -> 'a list | CCListLabels.tl : 'a list -> 'a list | BatList.tl : 'a list -> 'a list | Base.List.tl_exn : 'a list -> 'a list |
| | | | Base.List.tl : 'a list -> 'a list option |
| | | | Base.List.to_array : 'a list -> 'a array |
| | CCListLabels.to_gen : 'a list -> 'a CCListLabels.gen | | |
| | CCListLabels.to_iter : 'a list -> 'a CCListLabels.iter | | |
| | | | Base.List.to_list : 'a list -> 'a list |
| ListLabels.to_seq : 'a list -> 'a Seq.t | CCListLabels.to_seq : 'a list -> 'a Seq.t | | |
| | CCListLabels.to_string : ?start:string -> ?stop:string -> ?sep:string -> ('a -> string) -> 'a list -> string | | |
| | | | Base.List.transpose : 'a list list -> 'a list list option |
| | | BatList.transpose : 'a list list -> 'a list list | Base.List.transpose_exn : 'a list list -> 'a list list |
| | CCListLabels.union : eq:('a -> 'a -> bool) -> 'a list -> 'a list -> 'a list | | |
| | CCListLabels.uniq : eq:('a -> 'a -> bool) -> 'a list -> 'a list | | |
| | CCListLabels.uniq_succ : eq:('a -> 'a -> bool) -> 'a list -> 'a list | | |
| | | | Base.List.unordered_append : 'a list -> 'a list -> 'a list |
| | | | Base.List.unzip3 : ('a * 'b * 'c) list -> 'a list * 'b list * 'c list |
| | | | Base.List.zip : 'a list -> 'b list -> ('a * 'b) list Base.List.Or_unequal_lengths.t |
| | CCList.( -- ) : int -> int -> int list | | |
| | CCList.( --^ ) : int -> int -> int list | | |
| | CCList.( <$> ) : ('a -> 'b) -> 'a list -> 'b list | | |
| | CCList.( <*> ) : ('a -> 'b) list -> 'a list -> 'b list | | |
| | CCList.( >>= ) : 'a list -> ('a -> 'b list) -> 'b list | | |
| | CCList.( >\|= ) : 'a list -> ('a -> 'b) -> 'b list | | |
| | CCList.( @ ) : 'a list -> 'a list -> 'a list | | |
| | CCList.( and* ) : 'a list -> 'b list -> ('a * 'b) list | | |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| | CCList.( and+ ) : 'a list -> 'b list -> ('a * 'b) list | | |
| | CCList.( let* ) : 'a list -> ('a -> 'b list) -> 'b list | | |
| | CCList.( let+ ) : 'a list -> ('a -> 'b) -> 'b list | | |
| List.[] : 'a list = List.[] | CCList.[] : 'a list = [] | BatList.[] : 'a list = BatList.[] | |
| | CCList.Assoc.get : eq:('a -> 'a -> bool) -> 'a -> ('a, 'b) CCList.Assoc.t -> 'b option | | |
| | CCList.Assoc.get_exn : eq:('a -> 'a -> bool) -> 'a -> ('a, 'b) CCList.Assoc.t -> 'b | | |
| | CCList.Assoc.mem : ?eq:('a -> 'a -> bool) -> 'a -> ('a, 'b) CCList.Assoc.t -> bool | | |
| | CCList.Assoc.remove : eq:('a -> 'a -> bool) -> 'a -> ('a, 'b) CCList.Assoc.t -> ('a, 'b) CCList.Assoc.t | | |
| | CCList.Assoc.set : eq:('a -> 'a -> bool) -> 'a -> 'b -> ('a, 'b) CCList.Assoc.t -> ('a, 'b) CCList.Assoc.t | | |
| | CCList.Assoc.update : eq:('a -> 'a -> bool) -> f:('b option -> 'b option) -> 'a -> ('a, 'b) CCList.Assoc.t -> ('a, 'b) CCList.Assoc.t | | |
| | CCList.Infix.( and& ) : 'a list -> 'b list -> ('a * 'b) list | | |
| | CCList.add_nodup : eq:('a -> 'a -> bool) -> 'a -> 'a list -> 'a list | | |
| | CCList.all_ok : ('a, 'err) result list -> ('a CCList.t, 'err) result | | |
| | CCList.all_some : 'a option list -> 'a list option | | |
| List.append : 'a list -> 'a list -> 'a list | CCList.append : 'a list -> 'a list -> 'a list | BatList.append : 'a list -> 'a list -> 'a list | |
| List.assoc : 'a -> ('a * 'b) list -> 'b | | BatList.assoc : 'a -> ('a * 'b) list -> 'b | |
| | CCList.assoc : eq:('a -> 'a -> bool) -> 'a -> ('a * 'b) list -> 'b | | |
| | | BatList.assoc_inv : 'b -> ('a * 'b) list -> 'a | |
| List.assoc_opt : 'a -> ('a * 'b) list -> 'b option | | BatList.assoc_opt : 'a -> ('a * 'b) list -> 'b option | |
| | CCList.assoc_opt : eq:('a -> 'a -> bool) -> 'a -> ('a * 'b) list -> 'b option | | |
| List.assq : 'a -> ('a * 'b) list -> 'b | CCList.assq : 'a -> ('a * 'b) list -> 'b | BatList.assq : 'a -> ('a * 'b) list -> 'b | |
| | | BatList.assq_inv : 'b -> ('a * 'b) list -> 'a | |
| List.assq_opt : 'a -> ('a * 'b) list -> 'b option | CCList.assq_opt : 'a -> ('a * 'b) list -> 'b option | BatList.assq_opt : 'a -> ('a * 'b) list -> 'b option | |
| | | BatList.at : 'a list -> int -> 'a | |
| | | BatList.at_opt : 'a list -> int -> 'a option | |
| | | BatList.backwards : 'a list -> 'a BatEnum.t | |
| | CCList.cartesian_product : 'a list list -> 'a list list | BatList.n_cartesian_product : 'a list list -> 'a list list | |
| | | BatList.cartesian_product : 'a list -> 'b list -> ('a * 'b) list | |
| | CCList.chunks : int -> 'a list -> 'a list list | | |
| List.combine : 'a list -> 'b list -> ('a * 'b) list | CCList.combine : 'a list -> 'b list -> ('a * 'b) list | BatList.combine : 'a list -> 'b list -> ('a * 'b) list | |
| | CCList.combine_gen : 'a list -> 'b list -> ('a * 'b) CCList.gen | | |
| | CCList.combine_shortest : 'a list -> 'b list -> ('a * 'b) list | | |
| List.compare : ('a -> 'a -> int) -> 'a list -> 'a list -> int | CCList.compare : ('a -> 'a -> int) -> 'a list -> 'a list -> int | BatList.compare : 'a BatOrd.comp -> 'a list BatOrd.comp | |
| List.compare_length_with : 'a list -> int -> int | CCList.compare_length_with : 'a list -> int -> int | BatList.compare_length_with : 'a list -> int -> int | |
| List.compare_lengths : 'a list -> 'b list -> int | CCList.compare_lengths : 'a list -> 'b list -> int | BatList.compare_lengths : 'a list -> 'b list -> int | |
| List.concat : 'a list list -> 'a list | CCList.concat : 'a list list -> 'a list | BatList.concat : 'a list list -> 'a list | |
| List.concat_map : ('a -> 'b list) -> 'a list -> 'b list | CCList.concat_map : ('a -> 'b list) -> 'a list -> 'b list | BatList.concat_map : ('a -> 'b list) -> 'a list -> 'b list | |
| List.cons : 'a -> 'a list -> 'a list | CCList.cons : 'a -> 'a list -> 'a list | BatList.cons : 'a -> 'a list -> 'a list | |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| | CCList.cons' : 'a list -> 'a -> 'a list | | |
| | CCList.cons_maybe : 'a option -> 'a list -> 'a list | | |
| | CCList.count : ('a -> bool) -> 'a list -> int | BatList.count_matching : ('a -> bool) -> 'a list -> int | |
| | CCList.count_true_false : ('a -> bool) -> 'a list -> int * int | | |
| | CCList.diagonal : 'a list -> ('a * 'a) list | | |
| | CCList.drop : int -> 'a list -> 'a list | BatList.drop : int -> 'a list -> 'a list | |
| | CCList.drop_while : ('a -> bool) -> 'a list -> 'a list | BatList.drop_while : ('a -> bool) -> 'a list -> 'a list | |
| | | BatList.dropwhile : ('a -> bool) -> 'a list -> 'a list | |
| | CCList.empty : 'a list = [] | | |
| | | BatList.enum : 'a list -> 'a BatEnum.t | |
| | | BatList.eq : 'a BatOrd.eq -> 'a list BatOrd.eq | |
| List.equal : ('a -> 'a -> bool) -> 'a list -> 'a list -> bool | CCList.equal : ('a -> 'a -> bool) -> 'a list -> 'a list -> bool | BatList.equal : ('a -> 'a -> bool) -> 'a list -> 'a list -> bool | |
| List.exists : ('a -> bool) -> 'a list -> bool | CCList.exists : ('a -> bool) -> 'a list -> bool | BatList.exists : ('a -> bool) -> 'a list -> bool | |
| List.exists2 : ('a -> 'b -> bool) -> 'a list -> 'b list -> bool | CCList.exists2 : ('a -> 'b -> bool) -> 'a list -> 'b list -> bool | BatList.exists2 : ('a -> 'b -> bool) -> 'a list -> 'b list -> bool | |
| List.fast_sort : ('a -> 'a -> int) -> 'a list -> 'a list | CCList.fast_sort : ('a -> 'a -> int) -> 'a list -> 'a list | BatList.fast_sort : ('a -> 'a -> int) -> 'a list -> 'a list | |
| | | BatList.favg : float list -> float | |
| List.filter : ('a -> bool) -> 'a list -> 'a list | CCList.filter : ('a -> bool) -> 'a list -> 'a list | BatList.filter : ('a -> bool) -> 'a list -> 'a list | |
| List.filter_map : ('a -> 'b option) -> 'a list -> 'b list | CCList.filter_map : ('a -> 'b option) -> 'a list -> 'b list | BatList.filter_map : ('a -> 'b option) -> 'a list -> 'b list | |
| List.filteri : (int -> 'a -> bool) -> 'a list -> 'a list | CCList.filteri : (int -> 'a -> bool) -> 'a list -> 'a list | BatList.filteri : (int -> 'a -> bool) -> 'a list -> 'a list | |
| | | BatList.filteri_map : (int -> 'a -> 'b option) -> 'a list -> 'b list | |
| List.find : ('a -> bool) -> 'a list -> 'a | CCList.find : ('a -> bool) -> 'a list -> 'a | BatList.find : ('a -> bool) -> 'a list -> 'a | |
| List.find_all : ('a -> bool) -> 'a list -> 'a list | CCList.find_all : ('a -> bool) -> 'a list -> 'a list | BatList.find_all : ('a -> bool) -> 'a list -> 'a list | |
| | | BatList.find_exn : ('a -> bool) -> exn -> 'a list -> 'a | |
| | CCList.find_idx : ('a -> bool) -> 'a list -> (int * 'a) option | | |
| List.find_map : ('a -> 'b option) -> 'a list -> 'b option | CCList.find_map : ('a -> 'b option) -> 'a list -> 'b option | BatList.find_map_opt : ('a -> 'b option) -> 'a list -> 'b option | |
| | | BatList.find_map : ('a -> 'b option) -> 'a list -> 'b | |
| | CCList.find_mapi : (int -> 'a -> 'b option) -> 'a list -> 'b option | | |
| List.find_opt : ('a -> bool) -> 'a list -> 'a option | CCList.find_opt : ('a -> bool) -> 'a list -> 'a option | BatList.find_opt : ('a -> bool) -> 'a list -> 'a option | |
| | CCList.find_pred : ('a -> bool) -> 'a list -> 'a option | | |
| | CCList.find_pred_exn : ('a -> bool) -> 'a list -> 'a | | |
| | | BatList.findi : (int -> 'a -> bool) -> 'a list -> int * 'a | |
| | | BatList.first : 'a list -> 'a | |
| | CCList.flat_map : ('a -> 'b list) -> 'a list -> 'b list | | |
| | CCList.flat_map_i : (int -> 'a -> 'b list) -> 'a list -> 'b list | | |
| List.flatten : 'a list list -> 'a list | CCList.flatten : 'a list list -> 'a list | BatList.flatten : 'a list list -> 'a list | |
| | | BatList.fold : ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a | |
| | CCList.fold_filter_map : ('acc -> 'a -> 'acc * 'b option) -> 'acc -> 'a list -> 'acc * 'b list | | |
| | CCList.fold_filter_map_i : ('acc -> int -> 'a -> 'acc * 'b option) -> 'acc -> 'a list -> 'acc * 'b list | | |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| | CCList.fold_flat_map : ('acc -> 'a -> 'acc * 'b list) -> 'acc -> 'a list -> 'acc * 'b list | | |
| | CCList.fold_flat_map_i : ('acc -> int -> 'a -> 'acc * 'b list) -> 'acc -> 'a list -> 'acc * 'b list | | |
| List.fold_left : ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a | CCList.fold_left : ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a | BatList.fold_left : ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a | |
| List.fold_left2 : ('a -> 'b -> 'c -> 'a) -> 'a -> 'b list -> 'c list -> 'a | CCList.fold_left2 : ('a -> 'b -> 'c -> 'a) -> 'a -> 'b list -> 'c list -> 'a | BatList.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 | CCList.fold_left_map : ('a -> 'b -> 'a * 'c) -> 'a -> 'b list -> 'a * 'c list | BatList.fold_left_map : ('a -> 'b -> 'a * 'c) -> 'a -> 'b list -> 'a * 'c list | |
| | CCList.fold_map : ('acc -> 'a -> 'acc * 'b) -> 'acc -> 'a list -> 'acc * 'b list | | |
| | CCList.fold_map2 : ('acc -> 'a -> 'b -> 'acc * 'c) -> 'acc -> 'a list -> 'b list -> 'acc * 'c list | | |
| | CCList.fold_map_i : ('acc -> int -> 'a -> 'acc * 'b) -> 'acc -> 'a list -> 'acc * 'b list | | |
| | CCList.fold_on_map : f:('a -> 'b) -> reduce:('acc -> 'b -> 'acc) -> 'acc -> 'a list -> 'acc | | |
| | CCList.fold_product : ('c -> 'a -> 'b -> 'c) -> 'c -> 'a list -> 'b list -> 'c | | |
| List.fold_right : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b | CCList.fold_right : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b | BatList.fold_right : ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b | |
| List.fold_right2 : ('a -> 'b -> 'c -> 'c) -> 'a list -> 'b list -> 'c -> 'c | CCList.fold_right2 : ('a -> 'b -> 'c -> 'c) -> 'a list -> 'b list -> 'c -> 'c | BatList.fold_right2 : ('a -> 'b -> 'c -> 'c) -> 'a list -> 'b list -> 'c -> 'c | |
| | | BatList.fold_righti : (int -> 'b -> 'a -> 'a) -> 'b list -> 'a -> 'a | |
| | CCList.fold_while : ('a -> 'b -> 'a * [ `Continue \| `Stop ]) -> 'a -> 'b list -> 'a | | |
| | | BatList.fold_while : ('acc -> 'a -> bool) -> ('acc -> 'a -> 'acc) -> 'acc -> 'a list -> 'acc * 'a list | |
| | CCList.foldi : ('b -> int -> 'a -> 'b) -> 'b -> 'a list -> 'b | BatList.fold_lefti : ('a -> int -> 'b -> 'a) -> 'a -> 'b list -> 'a | |
| | CCList.foldi2 : ('c -> int -> 'a -> 'b -> 'c) -> 'c -> 'a list -> 'b list -> 'c | | |
| List.for_all : ('a -> bool) -> 'a list -> bool | CCList.for_all : ('a -> bool) -> 'a list -> bool | BatList.for_all : ('a -> bool) -> 'a list -> bool | |
| List.for_all2 : ('a -> 'b -> bool) -> 'a list -> 'b list -> bool | CCList.for_all2 : ('a -> 'b -> bool) -> 'a list -> 'b list -> bool | BatList.for_all2 : ('a -> 'b -> bool) -> 'a list -> 'b list -> bool | |
| | | BatList.frange : float -> [< `Downto \| `To ] -> float -> int -> float list | |
| | | BatList.fsum : float list -> float | |
| | CCList.get_at_idx : int -> 'a list -> 'a option | | |
| | CCList.get_at_idx_exn : int -> 'a list -> 'a | | |
| | | BatList.group : ('a -> 'a -> int) -> 'a list -> 'a list list | |
| | CCList.group_by : ?hash:('a -> int) -> ?eq:('a -> 'a -> bool) -> 'a list -> 'a list list | | |
| | | BatList.group_consecutive : ('a -> 'a -> bool) -> 'a list -> 'a list list | |
| | CCList.group_join_by : ?eq:('a -> 'a -> bool) -> ?hash:('a -> int) -> ('b -> 'a) -> 'a list -> 'b list -> ('a * 'b list) list | | |
| | CCList.group_succ : eq:('a -> 'a -> bool) -> 'a list -> 'a list list | | |
| List.hd : 'a list -> 'a | CCList.hd : 'a list -> 'a | BatList.hd : 'a list -> 'a | |
| | CCList.hd_tl : 'a list -> 'a * 'a list | | |
| | CCList.head_opt : 'a list -> 'a option | | |
| | | BatList.index_of : 'a -> 'a list -> int option | |
| | | BatList.index_ofq : 'a -> 'a list -> int option | |
| List.init : int -> (int -> 'a) -> 'a list | CCList.init : int -> (int -> 'a) -> 'a list | BatList.init : int -> (int -> 'a) -> 'a list | |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| | CCList.insert_at_idx : int -> 'a -> 'a list -> 'a list | | |
| | CCList.inter : eq:('a -> 'a -> bool) -> 'a list -> 'a list -> 'a list | | |
| | CCList.interleave : 'a list -> 'a list -> 'a list | | |
| | CCList.intersperse : 'a -> 'a list -> 'a list | BatList.interleave : ?first:'a -> ?last:'a -> 'a -> 'a list -> 'a list | |
| | CCList.is_empty : 'a list -> bool | BatList.is_empty : 'a list -> bool | |
| | CCList.is_sorted : cmp:('a -> 'a -> int) -> 'a list -> bool | | |
| List.iter : ('a -> unit) -> 'a list -> unit | CCList.iter : ('a -> unit) -> 'a list -> unit | BatList.iter : ('a -> unit) -> 'a list -> unit | |
| List.iter2 : ('a -> 'b -> unit) -> 'a list -> 'b list -> unit | CCList.iter2 : ('a -> 'b -> unit) -> 'a list -> 'b list -> unit | BatList.iter2 : ('a -> 'b -> unit) -> 'a list -> 'b list -> unit | |
| | | BatList.iter2i : (int -> 'a -> 'b -> unit) -> 'a list -> 'b list -> unit | |
| List.iteri : (int -> 'a -> unit) -> 'a list -> unit | CCList.iteri : (int -> 'a -> unit) -> 'a list -> unit | BatList.iteri : (int -> 'a -> unit) -> 'a list -> unit | |
| | CCList.iteri2 : (int -> 'a -> 'b -> unit) -> 'a list -> 'b list -> unit | | |
| | CCList.join : join_row:('a -> 'b -> 'c option) -> 'a list -> 'b list -> 'c list | | |
| | CCList.join_all_by : ?eq:('key -> 'key -> bool) -> ?hash:('key -> int) -> ('a -> 'key) -> ('b -> 'key) -> merge:('key -> 'a list -> 'b list -> 'c option) -> 'a list -> 'b list -> 'c list | | |
| | CCList.join_by : ?eq:('key -> 'key -> bool) -> ?hash:('key -> int) -> ('a -> 'key) -> ('b -> 'key) -> merge:('key -> 'a -> 'b -> 'c option) -> 'a list -> 'b list -> 'c list | | |
| | | BatList.kahan_sum : float list -> float | |
| | CCList.keep_ok : ('a, 'b) result list -> 'a list | | |
| | CCList.keep_some : 'a option list -> 'a list | | |
| | CCList.last : int -> 'a list -> 'a list | | |
| | | BatList.last : 'a list -> 'a | |
| | CCList.last_opt : 'a list -> 'a option | | |
| List.length : 'a list -> int | CCList.length : 'a list -> int | BatList.length : 'a list -> int | |
| | | BatList.make : int -> 'a -> 'a list | |
| List.map : ('a -> 'b) -> 'a list -> 'b list | CCList.map : ('a -> 'b) -> 'a list -> 'b list | BatList.map : ('a -> 'b) -> 'a list -> 'b list | |
| List.map2 : ('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | CCList.map2 : ('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | BatList.map2 : ('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | |
| | | BatList.map2i : (int -> 'a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | |
| | CCList.map_product_l : ('a -> 'b list) -> 'a list -> 'b list list | | |
| List.mapi : (int -> 'a -> 'b) -> 'a list -> 'b list | CCList.mapi : (int -> 'a -> 'b) -> 'a list -> 'b list | BatList.mapi : (int -> 'a -> 'b) -> 'a list -> 'b list | |
| | | BatList.max : 'a list -> 'a | |
| List.mem : 'a -> 'a list -> bool | CCList.mem : ?eq:('a -> 'a -> bool) -> 'a -> 'a list -> bool | BatList.mem : 'a -> 'a list -> bool | |
| List.mem_assoc : 'a -> ('a * 'b) list -> bool | CCList.mem_assoc : ?eq:('a -> 'a -> bool) -> 'a -> ('a * 'b) list -> bool | BatList.mem_assoc : 'a -> ('a * 'b) list -> bool | |
| List.mem_assq : 'a -> ('a * 'b) list -> bool | CCList.mem_assq : 'a -> ('a * 'b) list -> bool | BatList.mem_assq : 'a -> ('a * 'b) list -> bool | |
| | | BatList.mem_cmp : ('a -> 'a -> int) -> 'a -> 'a list -> bool | |
| List.memq : 'a -> 'a list -> bool | CCList.memq : 'a -> 'a list -> bool | BatList.memq : 'a -> 'a list -> bool | |
| List.merge : ('a -> 'a -> int) -> 'a list -> 'a list -> 'a list | CCList.merge : ('a -> 'a -> int) -> 'a list -> 'a list -> 'a list | BatList.merge : ('a -> 'a -> int) -> 'a list -> 'a list -> 'a list | |
| | CCList.mguard : bool -> unit list | | |
| | | BatList.min : 'a list -> 'a | |
| | | BatList.min_max : ?cmp:('a -> 'a -> int) -> 'a list -> 'a * 'a | |
| | | BatList.modify : 'a -> ('b -> 'b) -> ('a * 'b) list -> ('a * 'b) list | |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| | | BatList.modify_at : int -> ('a -> 'a) -> 'a list -> 'a list | |
| | | BatList.modify_def : 'b -> 'a -> ('b -> 'b) -> ('a * 'b) list -> ('a * 'b) list | |
| | | BatList.modify_opt : 'a -> ('b option -> 'b option) -> ('a * 'b) list -> ('a * 'b) list | |
| | | BatList.modify_opt_at : int -> ('a -> 'a option) -> 'a list -> 'a list | |
| | CCList.cartesian_product : 'a list list -> 'a list list | BatList.n_cartesian_product : 'a list list -> 'a list list | |
| | | BatList.nsplit : ('a -> bool) -> 'a list -> 'a list list | |
| | | BatList.ntake : int -> 'a list -> 'a list list | |
| List.nth : 'a list -> int -> 'a | CCList.nth : 'a list -> int -> 'a | BatList.nth : 'a list -> int -> 'a | |
| List.nth_opt : 'a list -> int -> 'a option | CCList.nth_opt : 'a list -> int -> 'a option | BatList.nth_opt : 'a list -> int -> 'a option | |
| | | BatList.of_backwards : 'a BatEnum.t -> 'a list | |
| | | BatList.of_enum : 'a BatEnum.t -> 'a list | |
| | CCList.of_gen : 'a CCList.gen -> 'a list | | |
| | CCList.of_iter : 'a CCList.iter -> 'a list | | |
| List.of_seq : 'a Seq.t -> 'a list | CCList.of_seq : 'a Seq.t -> 'a list | BatList.of_seq : 'a Seq.t -> 'a list | |
| | CCList.of_seq_rev : 'a Seq.t -> 'a list | | |
| | | BatList.ord : 'a BatOrd.ord -> 'a list BatOrd.ord | |
| List.partition : ('a -> bool) -> 'a list -> 'a list * 'a list | CCList.partition : ('a -> bool) -> 'a list -> 'a list * 'a list | BatList.partition : ('a -> bool) -> 'a list -> 'a list * 'a list | |
| | CCList.partition_filter_map : ('a -> [< `Drop \| `Left of 'b \| `Right of 'c ]) -> 'a list -> 'b list * 'c list | | |
| | CCList.partition_map : ('a -> [< `Drop \| `Left of 'b \| `Right of 'c ]) -> 'a list -> 'b list * 'c list | | |
| List.partition_map : ('a -> ('b, 'c) Either.t) -> 'a list -> 'b list * 'c list | CCList.partition_map_either : ('a -> ('b, 'c) CCEither.t) -> 'a list -> 'b list * 'c list | BatList.partition_map : ('a -> ('b, 'c) BatEither.t) -> 'a list -> 'b list * 'c list | |
| | CCList.pp : ?pp_start:unit CCList.printer -> ?pp_stop:unit CCList.printer -> ?pp_sep:unit CCList.printer -> 'a CCList.printer -> 'a list CCList.printer | | |
| | | BatList.print : ?first:string -> ?last:string -> ?sep:string -> ('a BatInnerIO.output -> 'b -> unit) -> 'a BatInnerIO.output -> 'b list -> unit | |
| | CCList.product : ('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | | |
| | CCList.pure : 'a -> 'a list | | |
| | CCList.random : 'a CCList.random_gen -> 'a list CCList.random_gen | | |
| | CCList.random_choose : 'a list -> 'a CCList.random_gen | | |
| | CCList.random_len : int -> 'a CCList.random_gen -> 'a list CCList.random_gen | | |
| | CCList.random_non_empty : 'a CCList.random_gen -> 'a list CCList.random_gen | | |
| | CCList.random_sequence : 'a CCList.random_gen list -> 'a list CCList.random_gen | | |
| | CCList.range : int -> int -> int list | | |
| | CCList.range' : int -> int -> int list | | |
| | | BatList.range : int -> [< `Downto \| `To ] -> int -> int list | |
| | CCList.range_by : step:int -> int -> int -> int list | | |
| | CCList.reduce : ('a -> 'a -> 'a) -> 'a list -> 'a option | | |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| | CCList.reduce_exn : ('a -> 'a -> 'a) -> 'a list -> 'a | BatList.reduce : ('a -> 'a -> 'a) -> 'a list -> 'a | |
| | CCList.remove : eq:('a -> 'a -> bool) -> key:'a -> 'a list -> 'a list | | |
| | | BatList.remove : 'a list -> 'a -> 'a list | |
| | | BatList.remove_all : 'a list -> 'a -> 'a list | |
| List.remove_assoc : 'a -> ('a * 'b) list -> ('a * 'b) list | CCList.remove_assoc : eq:('a -> 'a -> bool) -> 'a -> ('a * 'b) list -> ('a * 'b) list | BatList.remove_assoc : 'a -> ('a * 'b) list -> ('a * 'b) list | |
| List.remove_assq : 'a -> ('a * 'b) list -> ('a * 'b) list | CCList.remove_assq : 'a -> ('a * 'b) list -> ('a * 'b) list | BatList.remove_assq : 'a -> ('a * 'b) list -> ('a * 'b) list | |
| | CCList.remove_at_idx : int -> 'a list -> 'a list | BatList.remove_at : int -> 'a list -> 'a list | |
| | | BatList.remove_if : ('a -> bool) -> 'a list -> 'a list | |
| | CCList.remove_one : eq:('a -> 'a -> bool) -> 'a -> 'a list -> 'a list | | |
| | CCList.repeat : int -> 'a list -> 'a list | | |
| | CCList.replicate : int -> 'a -> 'a list | | |
| | CCList.return : 'a -> 'a list | | |
| List.rev : 'a list -> 'a list | CCList.rev : 'a list -> 'a list | BatList.rev : 'a list -> 'a list | |
| List.rev_append : 'a list -> 'a list -> 'a list | CCList.rev_append : 'a list -> 'a list -> 'a list | BatList.rev_append : 'a list -> 'a list -> 'a list | |
| List.rev_map : ('a -> 'b) -> 'a list -> 'b list | CCList.rev_map : ('a -> 'b) -> 'a list -> 'b list | BatList.rev_map : ('a -> 'b) -> 'a list -> 'b list | |
| List.rev_map2 : ('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | CCList.rev_map2 : ('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | BatList.rev_map2 : ('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list | |
| | | BatList.rfind : ('a -> bool) -> 'a list -> 'a | |
| | | BatList.rindex_of : 'a -> 'a list -> int option | |
| | | BatList.rindex_ofq : 'a -> 'a list -> int option | |
| | CCList.scan_left : ('acc -> 'a -> 'acc) -> 'acc -> 'a list -> 'acc list | | |
| | CCList.set_at_idx : int -> 'a -> 'a list -> 'a list | | |
| | | BatList.shuffle : ?state:Random.State.t -> 'a list -> 'a list | |
| | | BatList.singleton : 'a -> 'a list | |
| List.sort : ('a -> 'a -> int) -> 'a list -> 'a list | CCList.sort : ('a -> 'a -> int) -> 'a list -> 'a list | BatList.sort : ('a -> 'a -> int) -> 'a list -> 'a list | |
| List.sort_uniq : ('a -> 'a -> int) -> 'a list -> 'a list | CCList.sort_uniq : cmp:('a -> 'a -> int) -> 'a list -> 'a list | BatList.sort_uniq : ('a -> 'a -> int) -> 'a list -> 'a list | |
| | | BatList.sort_unique : ('a -> 'a -> int) -> 'a list -> 'a list | |
| | CCList.sorted_insert : cmp:('a -> 'a -> int) -> ?uniq:bool -> 'a -> 'a list -> 'a list | | |
| | CCList.sorted_merge : cmp:('a -> 'a -> int) -> 'a list -> 'a list -> 'a list | | |
| | CCList.sorted_merge_uniq : cmp:('a -> 'a -> int) -> 'a list -> 'a list -> 'a list | | |
| | | BatList.span : ('a -> bool) -> 'a list -> 'a list * 'a list | |
| List.split : ('a * 'b) list -> 'a list * 'b list | CCList.split : ('a * 'b) list -> 'a list * 'b list | BatList.split : ('a * 'b) list -> 'a list * 'b list | |
| | | BatList.split_at : int -> 'a list -> 'a list * 'a list | |
| | | BatList.split_nth : int -> 'a list -> 'a list * 'a list | |
| List.stable_sort : ('a -> 'a -> int) -> 'a list -> 'a list | CCList.stable_sort : ('a -> 'a -> int) -> 'a list -> 'a list | BatList.stable_sort : ('a -> 'a -> int) -> 'a list -> 'a list | |
| | CCList.sublists_of_len : ?last:('a list -> 'a list option) -> ?offset:int -> int -> 'a list -> 'a list list | | |
| | CCList.subset : eq:('a -> 'a -> bool) -> 'a list -> 'a list -> bool | | |
| | | BatList.subset : ('a -> 'b -> int) -> 'a list -> 'b list -> bool | |
| | | BatList.sum : int list -> int | |

| Stdlib | Containers | Batteries | Base |
|---|---|---|---|
| | CCList.tail_opt : 'a list -> 'a list option | | |
| | CCList.take : int -> 'a list -> 'a list | BatList.take : int -> 'a list -> 'a list | |
| | CCList.take_drop : int -> 'a list -> 'a list * 'a list | | |
| | CCList.take_drop_while : ('a -> bool) -> 'a list -> 'a list * 'a list | | |
| | CCList.take_while : ('a -> bool) -> 'a list -> 'a list | BatList.take_while : ('a -> bool) -> 'a list -> 'a list | |
| | | BatList.takedrop : int -> 'a list -> 'a list * 'a list | |
| | | BatList.takewhile : ('a -> bool) -> 'a list -> 'a list | |
| List.tl : 'a list -> 'a list | CCList.tl : 'a list -> 'a list | BatList.tl : 'a list -> 'a list | |
| | CCList.to_gen : 'a list -> 'a CCList.gen | | |
| | CCList.to_iter : 'a list -> 'a CCList.iter | | |
| List.to_seq : 'a list -> 'a Seq.t | CCList.to_seq : 'a list -> 'a Seq.t | BatList.to_seq : 'a list -> 'a Seq.t | |
| | CCList.to_string : ?start:string -> ?stop:string -> ?sep:string -> ('a -> string) -> 'a list -> string | | |
| | | BatList.transpose : 'a list list -> 'a list list | |
| | | BatList.unfold : 'b -> ('b -> ('a * 'b) option) -> 'a list | |
| | | BatList.unfold_exc : (unit -> 'a) -> 'a list * exn | |
| | | BatList.unfold_exn : (unit -> 'a) -> 'a list * exn | |
| | CCList.union : eq:('a -> 'a -> bool) -> 'a list -> 'a list -> 'a list | | |
| | CCList.uniq : eq:('a -> 'a -> bool) -> 'a list -> 'a list | BatList.unique : ?eq:('a -> 'a -> bool) -> 'a list -> 'a list | |
| | CCList.uniq_succ : eq:('a -> 'a -> bool) -> 'a list -> 'a list | | |
| | | BatList.unique_cmp : ?cmp:('a -> 'a -> int) -> 'a list -> 'a list | |
| | | BatList.unique_hash : ?hash:('a -> int) -> ?eq:('a -> 'a -> bool) -> 'a list -> 'a list | |