Stdlib	Containers	Batteries	Base
	CCStringLabels.(<) : string -> string -> bool		Base.String.(<) : string -> string -> bool
	CCStringLabels.(<=) : string -> string -> bool		Base.String.(<=) : string -> string -> bool
	CCStringLabels.(<>) : string -> string -> bool		Base.String.(<>) : string -> string -> bool
	CCStringLabels.(=) : string -> string -> bool		Base.String.(=) : string -> string -> bool
	CCStringLabels.(>) : string -> string -> bool		Base.String.(>) : string -> string -> bool
	CCStringLabels.(>=) : string -> string -> bool		Base.String.(>=) : string -> string -> bool
			Base.String.(^) : string -> string -> string
			Base.String.ascending : string -> string -> int
		BatString.backwards : string -> char BatEnum.t	
			Base.String.between : string -> low:string -> high:string -> bool
StringLabels.blit : src:string -> src_pos:int -> dst:bytes -> dst_pos:int -> len:int -> unit	CCStringLabels.blit : src:string -> src_pos:int -> dst:bytes -> dst_pos:int -> len:int -> unit	BatString.blit : string -> int -> bytes -> int -> int -> unit	
StringLabels.capitalize : string -> string	CCStringLabels.capitalize : string -> string	BatString.capitalize : string -> string	Base.String.capitalize : string -> string
StringLabels.capitalize_ascii : string -> string	CCStringLabels.capitalize_ascii : string -> string	BatString.capitalize_ascii : string -> string	
		BatString.chop : ?l:int -> ?r:int -> string -> string	
	CCStringLabels.chop_prefix : pre:string -> string -> string option		Base.String.chop_prefix : string -> prefix:string -> string option
			Base.String.chop_prefix_exn : string -> prefix:string -> string
			Base.String.chop_prefix_if_exists : string -> prefix:string -> string
	CCStringLabels.chop_suffix : suf:string -> string -> string option		Base.String.chop_suffix : string -> suffix:string -> string option
			Base.String.chop_suffix_exn : string -> suffix:string -> string
			Base.String.chop_suffix_if_exists : string -> suffix:string -> string
			Base.String.clamp : string -> min:string -> max:string -> string Base.Or_error.t
			Base.String.clamp_exn : string -> min:string -> max:string -> string
			Base.String.comparator : (string, Base.String.comparator_witness) Base.Comparator.comparator = {Base.Comparator.compare; sexp_of_t}
StringLabels.compare : string -> string -> int	CCStringLabels.compare : string -> string -> int	BatString.compare : string -> string -> int	Base.String.compare : string -> string -> int
	CCStringLabels.compare_natural : string -> string -> int		
	CCStringLabels.compare_versions : string -> string -> int		
StringLabels.concat : sep:string -> string list -> string	CCStringLabels.concat : sep:string -> string list -> string	BatString.concat : string -> string list -> string	Base.String.concat : ?sep:string -> string list -> string
			Base.String.concat_array : ?sep:string -> string array -> string
	CCStringLabels.concat_gen : sep:string -> string CCStringLabels.gen -> string		
	CCStringLabels.concat_iter : sep:string -> string CCStringLabels.iter -> string		
	CCStringLabels.concat_seq : sep:string -> string Seq.t -> string		
StringLabels.contains : string -> char -> bool	CCStringLabels.contains : string -> char -> bool	BatString.contains : string -> char -> bool	Base.String.contains : ?pos:int -> ?len:int -> string -> char -> bool
StringLabels.contains_from : string -> int -> char -> bool	CCStringLabels.contains_from : string -> int -> char -> bool	BatString.contains_from : string -> int -> char -> bool	
StringLabels.copy : string -> string	CCStringLabels.copy : string -> string	BatString.copy : string -> string	Base.String.copy : string -> string
			Base.String.count : string -> f:(Base.String.elt -> bool) -> int
		BatString.count_char : string -> char -> int	
		BatString.count_string : string -> string -> int	

Stdlib	Containers	Batteries	Base
StringLabels.create : int -> bytes	CCStringLabels.create : int -> bytes	BatString.create : int -> bytes	
		BatString.cut_on_char : char -> int -> string -> string	
			Base.String.descending : string -> string -> int
	CCStringLabels.drop : int -> string -> string		
			Base.String.drop_prefix : string -> int -> string
			Base.String.drop_suffix : string -> int -> string
	CCStringLabels.drop_while : f:(char -> bool) -> string -> string		
	CCStringLabels.edit_distance : ?cutoff:int -> string -> string -> int	BatString.edit_distance : string -> string -> int	
		BatString.ends_with : string -> string -> bool	
		BatString.enum : string -> char BatEnum.t	
StringLabels.equal : string -> string -> bool	CCStringLabels.equal : string -> string -> bool	BatString.equal : string -> string -> bool	Base.String.equal : string -> string -> bool
	CCStringLabels.equal_caseless : string -> string -> bool		
StringLabels.escaped : string -> string	CCStringLabels.escaped : string -> string	BatString.escaped : string -> string	Base.String.escaped : string -> string
	CCStringLabels.exists : f:(char -> bool) -> string -> bool		Base.String.exists : string -> f:(Base.String.elt -> bool) -> bool
		BatString.exists : string -> string -> bool	
	CCStringLabels.exists2 : f:(char -> char -> bool) -> string -> string -> bool		
		BatString.explode : string -> char list	
StringLabels.fill : bytes -> pos:int -> len:int -> char -> unit	CCStringLabels.fill : bytes -> pos:int -> len:int -> char -> unit	BatString.fill : bytes -> int -> int -> char -> unit	
	CCStringLabels.filter : f:(char -> bool) -> string -> string	BatString.filter : (char -> bool) -> string -> string	Base.String.filter : string -> f:(char -> bool) -> string
	CCStringLabels.filter_map : f:(char -> char option) -> string -> string	BatString.filter_map : (char -> char option) -> string -> string	
	CCStringLabels.find : ?start:int -> sub:string -> string -> int	BatString.find : string -> string -> int	
			Base.String.find : string -> f:(Base.String.elt -> bool) -> Base.String.elt option
	CCStringLabels.find_all : ?start:int -> sub:string -> string -> int CCStringLabels.gen	BatString.find_all : string -> string -> int BatEnum.t	
	CCStringLabels.find_all_l : ?start:int -> sub:string -> string -> int list		
		BatString.find_from : string -> int -> string -> int	
			Base.String.find_map : string -> f:(Base.String.elt -> 'a option) -> 'a option
	CCStringLabels.flat_map : ?sep:string -> f:(char -> string) -> string -> string		Base.String.concat_map : ?sep:string -> string -> f:(char -> string) -> string
	CCStringLabels.fold : f:('a -> char -> 'a) -> init:'a -> string -> 'a	BatString.fold_left : ('a -> char -> 'a) -> 'a -> string -> 'a	Base.String.fold : string -> init:'accum -> f:('accum -> Base.String.elt -> 'accum) -> 'accum
	CCStringLabels.fold2 : f:('a -> char -> char -> 'a) -> init:'a -> string -> string -> 'a		
		BatString.fold_lefti : ('a -> int -> char -> 'a) -> 'a -> string -> 'a	
			Base.String.fold_result : string -> init:'accum -> f:('accum -> Base.String.elt -> ('accum, 'e) Base.Result.t) -> ('accum, 'e) Base.Result.t
		BatString.fold_right : (char -> 'a -> 'a) -> string -> 'a -> 'a	
		BatString.fold_righti : (int -> char -> 'a -> 'a) -> string -> 'a -> 'a	
			Base.String.fold_until : string -> init:'accum -> f:('accum -> Base.String.elt -> ('accum, 'final)

Stdlib	Containers	Batteries	Base
			Base.Container_intf.Continue_or_stop.t) -> finish:('accum -> 'final) -> 'final
	CCStringLabels.foldi : f:('a -> int -> char -> 'a) -> 'a -> string -> 'a		Base.String.foldi : string -> init:'a -> f:(int -> 'a -> char -> 'a) -> 'a
	CCStringLabels.for_all : f:(char -> bool) -> string -> bool		Base.String.for_all : string -> f:(Base.String.elt -> bool) -> bool
	CCStringLabels.for_all2 : f:(char -> char -> bool) -> string -> string -> bool		
StringLabels.get : string -> int -> char	CCStringLabels.get : string -> int -> char	BatString.get : string -> int -> char	Base.String.get : string -> int -> char
	CCStringLabels.hash : string -> int		Base.String.hash : string -> int
			Base.String.hash_fold_t : Base.Ppx_hash_lib.Std.Hash.state -> string -> Base.Ppx_hash_lib.Std.Hash.state
		BatString.head : string -> int -> string	
		BatString.icompare : string -> string -> int	
		BatString.implode : char list -> string	
		BatString.in_place_mirror : bytes -> unit	
StringLabels.index : string -> char -> int	CCStringLabels.index : string -> char -> int	BatString.index : string -> char -> int	Base.String.index_exn : string -> char -> int
			Base.String.index : string -> char -> int option
		BatString.index_after_n : char -> int -> string -> int	
StringLabels.index_from : string -> int -> char -> int	CCStringLabels.index_from : string -> int -> char -> int	BatString.index_from : string -> int -> char -> int	Base.String.index_from_exn : string -> int -> char -> int
			Base.String.index_from : string -> int -> char -> int option
StringLabels.index_from_opt : string -> int -> char -> int option	CCStringLabels.index_from_opt : string -> int -> char -> int option	BatString.index_from_opt : string -> int -> char -> int option	
StringLabels.index_opt : string -> char -> int option	CCStringLabels.index_opt : string -> char -> int option	BatString.index_opt : string -> char -> int option	
StringLabels.init : int -> f:(int -> char) -> string	CCStringLabels.init : int -> f:(int -> char) -> string	BatString.init : int -> (int -> char) -> string	Base.String.init : int -> f:(int -> char) -> string
			Base.String.invariant : string Base.Invariant_intf.inv
	CCStringLabels.is_empty : string -> bool	BatString.is_empty : string -> bool	Base.String.is_empty : string -> bool
			Base.String.is_prefix : string -> prefix:string -> bool
	CCStringLabels.is_sub : sub:string -> sub_pos:int -> string -> pos:int -> sub_len:int -> bool		
			Base.String.is_substring : string -> substring:string -> bool
			Base.String.is_substring_at : string -> pos:int -> substring:string -> bool
			Base.String.is_suffix : string -> suffix:string -> bool
StringLabels.iter : f:(char -> unit) -> string -> unit	CCStringLabels.iter : f:(char -> unit) -> string -> unit	BatString.iter : (char -> unit) -> string -> unit	Base.String.iter : string -> f:(Base.String.elt -> unit) -> unit
	CCStringLabels.iter2 : f:(char -> char -> unit) -> string -> string -> unit		
StringLabels.iteri : f:(int -> char -> unit) -> string -> unit	CCStringLabels.iteri : f:(int -> char -> unit) -> string -> unit	BatString.iteri : (int -> char -> unit) -> string -> unit	
	CCStringLabels.iteri2 : f:(int -> char -> char -> unit) -> string -> string -> unit		
		BatString.join : string -> string list -> string	
		BatString.lchop : ?n:int -> string -> string	
		BatString.left : string -> int -> string	
StringLabels.length : string -> int	CCStringLabels.length : string -> int	BatString.length : string -> int	Base.String.length : string -> int
			Base.String.lfindi : ?pos:int -> string -> f:(int -> char -> bool) -> int option
	CCStringLabels.lines : string -> string list		

Stdlib	Containers	Batteries	Base
	CCStringLabels.lines_gen : string -> string CCStringLabels.gen		
	CCStringLabels.lines_iter : string -> string CCStringLabels.iter		
	CCStringLabels.lines_seq : string -> string Seq.t		
StringLabels.lowercase : string -> string	CCStringLabels.lowercase : string -> string	BatString.lowercase : string -> string	Base.String.lowercase : string -> string
StringLabels.lowercase_ascii : string -> string	CCStringLabels.lowercase_ascii : string -> string	BatString.lowercase_ascii : string -> string	
			Base.String.lsplit2 : string -> on:char -> (string * string) option
			Base.String.lsplit2_exn : string -> on:char -> string * string
			Base.String.lstrip : ?drop:(char -> bool) -> string -> string
	CCStringLabels.ltrim : string -> string		
StringLabels.make : int -> char -> string	CCStringLabels.make : int -> char -> string	BatString.make : int -> char -> string	Base.String.make : int -> char -> string
StringLabels.map : f:(char -> char) -> string -> string	CCStringLabels.map : f:(char -> char) -> string -> string	BatString.map : (char -> char) -> string -> string	Base.String.map : string -> f:(char -> char) -> string
	CCStringLabels.map2 : f:(char -> char -> char) -> string -> string -> string		
StringLabels.mapi : f:(int -> char -> char) -> string -> string	CCStringLabels.mapi : f:(int -> char -> char) -> string -> string	BatString.mapi : (int -> char -> char) -> string -> string	Base.String.mapi : string -> f:(int -> char -> char) -> string
			Base.String.max : string -> string -> string
			Base.String.max_elt : string -> compare:(Base.String.elt -> Base.String.elt -> int) -> Base.String.elt option
			Base.String.max_length : int = 144115188075855863
	CCStringLabels.mem : ?start:int -> sub:string -> string -> bool		Base.String.mem : string -> Base.String.elt -> bool
			Base.String.min : string -> string -> string
			Base.String.min_elt : string -> compare:(Base.String.elt -> Base.String.elt -> int) -> Base.String.elt option
		BatString.nreplace : str:string -> sub:string -> by:string -> string	
		BatString.nsplit : string -> by:string -> string list	
		BatString.numeric_compare : string -> string -> int	
	CCStringLabels.of_array : char array -> string		
		BatString.of_backwards : char BatEnum.t -> string	
	CCStringLabels.of_char : char -> string	BatString.of_char : char -> string	Base.String.of_char : char -> string
			Base.String.of_char_list : char list -> string
		BatString.of_enum : char BatEnum.t -> string	
		BatString.of_float : float -> string	
	CCStringLabels.of_gen : char CCStringLabels.gen -> string		
		BatString.of_int : int -> string	
	CCStringLabels.of_iter : char CCStringLabels.iter -> string		
	CCStringLabels.of_list : char list -> string	BatString.of_list : char list -> string	
StringLabels.of_seq : char Seq.t -> string	CCStringLabels.of_seq : char Seq.t -> string	BatString.of_seq : char Seq.t -> string	
			Base.String.of_string : string -> string
		BatString.ord : string -> string -> BatOrd.order	
	CCStringLabels.pad : ?side:[`Left | `Right] -> ?c:char -> int -> string -> string		
	CCStringLabels.pp : Format.formatter -> string -> unit		Base.String.pp : Base.Formatter.t -> string -> unit
	CCStringLabels.pp_buf : Buffer.t -> string -> unit		

Stdlib	Containers	Batteries	Base
	CCStringLabels.prefix : pre:string -> string -> bool		
			Base.String.prefix : string -> int -> string
		BatString.print : 'a BatInnerIO.output -> string -> unit	
		BatString.print_quoted : 'a BatInnerIO.output -> string -> unit	
		BatString.println : 'a BatInnerIO.output -> string -> unit	
		BatString.quote : string -> string	
		BatString.rchop : ?n:int -> string -> string	
StringLabels.rcontains_from : string -> int -> char -> bool	CCStringLabels.rcontains_from : string -> int -> char -> bool	BatString.rcontains_from : string -> int -> char -> bool	
	CCStringLabels.rdrop_while : f:(char -> bool) -> string -> string		
	CCStringLabels.repeat : string -> int -> string	BatString.repeat : string -> int -> string	
	CCStringLabels.replace : ?which:[`All | `Left | `Right] -> sub:string -> by:string -> string -> string		
		BatString.replace : str:string -> sub:string -> by:string -> bool * string	
		BatString.replace_chars : (char -> string) -> string -> string	
	CCStringLabels.rev : string -> string	BatString.rev : string -> string	Base.String.rev : string -> string
		BatString.rev_in_place : bytes -> unit	
	CCStringLabels.rfind : sub:string -> string -> int	BatString.rfind : string -> string -> int	
			Base.String.rfindi : ?pos:int -> string -> f:(int -> char -> bool) -> int option
		BatString.rfind_from : string -> int -> string -> int	
		BatString.right : string -> int -> string	
StringLabels.rindex : string -> char -> int	CCStringLabels.rindex : string -> char -> int	BatString.rindex : string -> char -> int	Base.String.rindex_exn : string -> char -> int
			Base.String.rindex : string -> char -> int option
StringLabels.rindex_from : string -> int -> char -> int	CCStringLabels.rindex_from : string -> int -> char -> int	BatString.rindex_from : string -> int -> char -> int	Base.String.rindex_from_exn : string -> int -> char -> int
			Base.String.rindex_from : string -> int -> char -> int option
StringLabels.rindex_from_opt : string -> int -> char -> int option	CCStringLabels.rindex_from_opt : string -> int -> char -> int option	BatString.rindex_from_opt : string -> int -> char -> int option	
StringLabels.rindex_opt : string -> char -> int option	CCStringLabels.rindex_opt : string -> char -> int option	BatString.rindex_opt : string -> char -> int option	
		BatString.rsplit : string -> by:string -> string * string	
			Base.String.rsplit2 : string -> on:char -> (string * string) option
			Base.String.rsplit2_exn : string -> on:char -> string * string
			Base.String.rstrip : ?drop:(char -> bool) -> string -> string
	CCStringLabels.rtrim : string -> string		
StringLabels.set : bytes -> int -> char -> unit	CCStringLabels.set : string -> int -> char -> string	BatString.set : bytes -> int -> char -> unit	
			Base.String.sexp_of_t : string -> Sexplib0__.Sexp.t
		BatString.slice : ?first:int -> ?last:int -> string -> string	
		BatString.splice : string -> int -> int -> string ->	

Stdlib	Containers	Batteries	Base
		string	
	CCStringLabels.split : by:string -> string -> string list		
		BatString.split : string -> by:string -> string * string	
StringLabels.split_on_char : sep:char -> string -> string list	CCStringLabels.split_on_char : by:char -> string -> string list	BatString.split_on_char : char -> string -> string list	Base.String.split : string -> on:char -> string list
			Base.String.split_lines : string -> string list
			Base.String.split_on_chars : string -> on:char list -> string list
		BatString.split_on_string : by:string -> string -> string list	
		BatString.starts_with : string -> string -> bool	
		BatString.strip : ?chars:string -> string -> string	Base.String.strip : ?drop:(char -> bool) -> string -> string
StringLabels.sub : string -> pos:int -> len:int -> string	CCStringLabels.sub : string -> pos:int -> len:int -> string	BatString.sub : string -> int -> int -> string	Base.String.sub : (string, string) Base.Blit.sub
			Base.String.subo : (string, string) Base.Blit.subo
			Base.String.substr_index : ?pos:int -> string -> pattern:string -> int option
			Base.String.substr_index_all : string -> may_overlap:bool -> pattern:string -> int list
			Base.String.substr_index_exn : ?pos:int -> string -> pattern:string -> int
			Base.String.substr_replace_all : string -> pattern:string -> with_:string -> string
			Base.String.substr_replace_first : ?pos:int -> string -> pattern:string -> with_:string -> string
	CCStringLabels.suffix : suf:string -> string -> bool		
			Base.String.suffix : string -> int -> string
			Base.String.sum : (module Base.Container_intf.Summable with type t = 'sum) -> string -> f: (Base.String.elt -> 'sum) -> 'sum
			Base.String.t_of_sexp : Sexplib0__.Sexp.t -> string
			Base.String.t_sexp_grammar : Base.Ppx_sexp_conv_lib.Sexp.Private.Raw_grammar.t...
		BatString.tail : string -> int -> string	
	CCStringLabels.take : int -> string -> string		
	CCStringLabels.take_drop : int -> string -> string * string		
	CCStringLabels.to_array : string -> char array		Base.String.to_array : string -> Base.String.elt array
		BatString.to_float : string -> float	
	CCStringLabels.to_gen : string -> char CCStringLabels.gen		
		BatString.to_int : string -> int	
	CCStringLabels.to_iter : string -> char CCStringLabels.iter		
	CCStringLabels.to_list : string -> char list	BatString.to_list : string -> char list	Base.String.to_list : string -> Base.String.elt list
			Base.String.to_list_rev : string -> char list
StringLabels.to_seq : string -> char Seq.t	CCStringLabels.to_seq : string -> char Seq.t	BatString.to_seq : string -> char Seq.t	
StringLabels.to_seqi : string -> (int * char) Seq.t	CCStringLabels.to_seqi : string -> (int * char) Seq.t	BatString.to_seqi : string -> (int * char) Seq.t	
			Base.String.to_string : string -> string
			Base.String.tr : target:char -> replacement:char -> string -> string
			Base.String.tr_multi : target:string -> replacement:string -> (string -> string) Base.Staged.t
StringLabels.trim : string -> string	CCStringLabels.trim : string -> string	BatString.trim : string -> string	
StringLabels.uncapitalize : string -> string	CCStringLabels.uncapitalize : string -> string	BatString.uncapitalize : string -> string	Base.String.uncapitalize : string -> string
StringLabels.uncapitalize_ascii : string -> string	CCStringLabels.uncapitalize_ascii : string -> string	BatString.uncapitalize_ascii : string -> string	

Stdlib	Containers	Batteries	Base
	CCStringLabels.unlines : string list -> string		
	CCStringLabels.unlines_gen : string CCStringLabels.gen -> string		
	CCStringLabels.unlines_iter : string CCStringLabels.iter -> string		
	CCStringLabels.unlines_seq : string Seq.t -> string		
StringLabels.unsafe_blit : src:string -> src_pos:int -> dst:bytes -> dst_pos:int -> len:int -> unit	CCStringLabels.unsafe_blit : src:string -> src_pos:int -> dst:bytes -> dst_pos:int -> len:int -> unit	BatString.unsafe_blit : string -> int -> bytes -> int -> int -> unit	
StringLabels.unsafe_fill : bytes -> pos:int -> len:int -> char -> unit	CCStringLabels.unsafe_fill : bytes -> pos:int -> len:int -> char -> unit	BatString.unsafe_fill : bytes -> int -> int -> char -> unit	
StringLabels.unsafe_get : string -> int -> char	CCStringLabels.unsafe_get : string -> int -> char	BatString.unsafe_get : string -> int -> char	Base.String.unsafe_get : string -> int -> char
StringLabels.unsafe_set : bytes -> int -> char -> unit	CCStringLabels.unsafe_set : bytes -> int -> char -> unit	BatString.unsafe_set : bytes -> int -> char -> unit	
StringLabels.uppercase : string -> string	CCStringLabels.uppercase : string -> string	BatString.uppercase : string -> string	Base.String.uppercase : string -> string
StringLabels.uppercase_ascii : string -> string	CCStringLabels.uppercase_ascii : string -> string	BatString.uppercase_ascii : string -> string	
			Base.String.validate_bound : min:string Base.Maybe_bound.t -> max:string Base.Maybe_bound.t -> string Base.Validate.check
			Base.String.validate_lbound : min:string Base.Maybe_bound.t -> string Base.Validate.check
			Base.String.validate_ubound : max:string Base.Maybe_bound.t -> string Base.Validate.check
	CCString.(<) : string -> string -> bool		
	CCString.(<=) : string -> string -> bool		
	CCString.(<>) : string -> string -> bool		
	CCString.(=) : string -> string -> bool		
	CCString.(>) : string -> string -> bool		
	CCString.(>=) : string -> string -> bool		
String.blit : string -> int -> bytes -> int -> int -> unit	CCString.blit : string -> int -> bytes -> int -> int -> unit		
String.capitalize : string -> string	CCString.capitalize : string -> string		
String.capitalize_ascii : string -> string	CCString.capitalize_ascii : string -> string		
	CCString.chop_prefix : pre:string -> string -> string option		
	CCString.chop_suffix : suf:string -> string -> string option		
String.compare : String.t -> String.t -> int	CCString.compare : string -> string -> int		
	CCString.compare_natural : string -> string -> int		
	CCString.compare_versions : string -> string -> int		
String.concat : string -> string list -> string	CCString.concat : string -> string list -> string		
	CCString.concat_gen : sep:string -> string CCString.gen -> string		
	CCString.concat_iter : sep:string -> string CCString.iter -> string		
	CCString.concat_seq : sep:string -> string Seq.t -> string		
String.contains : string -> char -> bool	CCString.contains : string -> char -> bool		
String.contains_from : string -> int -> char -> bool	CCString.contains_from : string -> int -> char -> bool		
String.copy : string -> string	CCString.copy : string -> string		
String.create : int -> bytes	CCString.create : int -> bytes		
	CCString.drop : int -> string -> string		
	CCString.drop_while : (char -> bool) -> string -> string		

Stdlib	Containers	Batteries	Base
	CCString.edit_distance : ?cutoff:int -> string -> string -> int		
String.equal : String.t -> String.t -> bool	CCString.equal : string -> string -> bool		
	CCString.equal_caseless : string -> string -> bool		
String.escaped : string -> string	CCString.escaped : string -> string		
	CCString.exists : (char -> bool) -> string -> bool		
	CCString.exists2 : (char -> char -> bool) -> string -> string -> bool		
String.fill : bytes -> int -> int -> char -> unit	CCString.fill : bytes -> int -> int -> char -> unit		
	CCString.filter : (char -> bool) -> string -> string		
	CCString.filter_map : (char -> char option) -> string -> string		
	CCString.find : ?start:int -> sub:string -> string -> int		
	CCString.find_all : ?start:int -> sub:string -> string -> int CCString.gen		
	CCString.find_all_l : ?start:int -> sub:string -> string -> int list		
	CCString.flat_map : ?sep:string -> (char -> string) -> string -> string		
	CCString.fold : ('a -> char -> 'a) -> 'a -> string -> 'a		
	CCString.fold2 : ('a -> char -> char -> 'a) -> 'a -> string -> string -> 'a		
	CCString.foldi : ('a -> int -> char -> 'a) -> 'a -> string -> 'a		
	CCString.for_all : (char -> bool) -> string -> bool		
	CCString.for_all2 : (char -> char -> bool) -> string -> string -> bool		
String.get : string -> int -> char	CCString.get : string -> int -> char		
	CCString.hash : string -> int		
String.index : string -> char -> int	CCString.index : string -> char -> int		
String.index_from : string -> int -> char -> int	CCString.index_from : string -> int -> char -> int		
String.index_from_opt : string -> int -> char -> int option	CCString.index_from_opt : string -> int -> char -> int option		
String.index_opt : string -> char -> int option	CCString.index_opt : string -> char -> int option		
String.init : int -> (int -> char) -> string	CCString.init : int -> (int -> char) -> string		
	CCString.is_empty : string -> bool		
	CCString.is_sub : sub:string -> int -> string -> int -> sub_len:int -> bool		
String.iter : (char -> unit) -> string -> unit	CCString.iter : (char -> unit) -> string -> unit		
	CCString.iter2 : (char -> char -> unit) -> string -> string -> unit		
String.iteri : (int -> char -> unit) -> string -> unit	CCString.iteri : (int -> char -> unit) -> string -> unit		
	CCString.iteri2 : (int -> char -> char -> unit) -> string -> string -> unit		
String.length : string -> int	CCString.length : string -> int		
	CCString.lines : string -> string list		
	CCString.lines_gen : string -> string CCString.gen		
	CCString.lines_iter : string -> string CCString.iter		
	CCString.lines_seq : string -> string Seq.t		
String.lowercase : string -> string	CCString.lowercase : string -> string		

Stdlib	Containers	Batteries	Base
String.lowercase_ascii : string -> string	CCString.lowercase_ascii : string -> string		
	CCString.ltrim : string -> string		
String.make : int -> char -> string	CCString.make : int -> char -> string		
String.map : (char -> char) -> string -> string	CCString.map : (char -> char) -> string -> string		
	CCString.map2 : (char -> char -> char) -> string -> string -> string		
String.mapi : (int -> char -> char) -> string -> string	CCString.mapi : (int -> char -> char) -> string -> string		
	CCString.mem : ?start:int -> sub:string -> string -> bool		
	CCString.of_array : char array -> string		
	CCString.of_char : char -> string		
	CCString.of_gen : char CCString.gen -> string		
	CCString.of_iter : char CCString.iter -> string		
	CCString.of_list : char list -> string		
String.of_seq : char Seq.t -> String.t	CCString.of_seq : char Seq.t -> string		
	CCString.pad : ?side:[`Left | `Right] -> ?c:char -> int -> string -> string		
	CCString.pp : Format.formatter -> string -> unit		
	CCString.pp_buf : Buffer.t -> string -> unit		
	CCString.prefix : pre:string -> string -> bool		
String.rcontains_from : string -> int -> char -> bool	CCString.rcontains_from : string -> int -> char -> bool		
	CCString.rdrop_while : (char -> bool) -> string -> string		
	CCString.repeat : string -> int -> string		
	CCString.replace : ?which:[`All | `Left | `Right] -> sub:string -> by:string -> string -> string		
	CCString.rev : string -> string		
	CCString.rfind : sub:string -> string -> int		
String.rindex : string -> char -> int	CCString.rindex : string -> char -> int		
String.rindex_from : string -> int -> char -> int	CCString.rindex_from : string -> int -> char -> int		
String.rindex_from_opt : string -> int -> char -> int option	CCString.rindex_from_opt : string -> int -> char -> int option		
String.rindex_opt : string -> char -> int option	CCString.rindex_opt : string -> char -> int option		
	CCString.rtrim : string -> string		
String.set : bytes -> int -> char -> unit	CCString.set : string -> int -> char -> string		
	CCString.split : by:string -> string -> string list		
String.split_on_char : char -> string -> string list	CCString.split_on_char : char -> string -> string list		
String.sub : string -> int -> int -> string	CCString.sub : string -> int -> int -> string		
	CCString.suffix : suf:string -> string -> bool		
	CCString.take : int -> string -> string		
	CCString.take_drop : int -> string -> string * string		
	CCString.to_array : string -> char array		
	CCString.to_gen : string -> char CCString.gen		
	CCString.to_iter : string -> char CCString.iter		
	CCString.to_list : string -> char list		

Stdlib	Containers	Batteries	Base
String.to_seq : String.t -> char Seq.t	CCString.to_seq : string -> char Seq.t		
String.to_seqi : String.t -> (int * char) Seq.t	CCString.to_seqi : string -> (int * char) Seq.t		
String.trim : string -> string	CCString.trim : string -> string		
String.uncapitalize : string -> string	CCString.uncapitalize : string -> string		
String.uncapitalize_ascii : string -> string	CCString.uncapitalize_ascii : string -> string		
	CCString.uniq : (char -> char -> bool) -> string -> string		
	CCString.unlines : string list -> string		
	CCString.unlines_gen : string CCString.gen -> string		
	CCString.unlines_iter : string CCString.iter -> string		
	CCString.unlines_seq : string Seq.t -> string		
String.unsafe_blit : string -> int -> bytes -> int -> int -> unit	CCString.unsafe_blit : string -> int -> bytes -> int -> int -> unit		
String.unsafe_fill : bytes -> int -> int -> char -> unit	CCString.unsafe_fill : bytes -> int -> int -> char -> unit		
String.unsafe_get : string -> int -> char	CCString.unsafe_get : string -> int -> char		
String.unsafe_set : bytes -> int -> char -> unit	CCString.unsafe_set : bytes -> int -> char -> unit		
String.uppercase : string -> string	CCString.uppercase : string -> string		
String.uppercase_ascii : string -> string	CCString.uppercase_ascii : string -> string		