Ocaml cheat sheet

Threads

val exit : unit -> unit
val delay : float -> unit
val join : t -> unit

Modules

Define signature of module -> module type <modname> = sig include .. type ..val .. end (Do when it says define a signature)
Module <name> : modname with type … = <type> = struct .. type … let … end

Functors

Module <name> (X: <othermod>) :<mod> with type .. and type .. = struct …end (Implement a functor, arguments <othermods> are signatures)
Module <name> = <functorname> (modarg) (modarg2) (use <functor> to define module)

Files

Output

val output_string : out_channel -> string -> unit
val out_channel_length : out_channel -> int #chars in file
val fprintf : out_channel -> ('a, out_channel, unit) format -> 'a
val printf : ('a, out_channel, unit) format -> 'a like fprintf but prints to stdout
val sprintf : ('a, unit, string) format -> 'a formats the string but returns the string

Input

val input_line : in_channel -> string reads from file, raises End_of_file
val in_channel_length : in_channel -> int
val bscanf : Scanning.in_channel -> ('a, 'b, 'c, 'd) scanner 1 arg is file, second is stringformat

CLOSE THE FILES!!!

Strings

val make : int -> char -> string string of length n holding char c
val get : string -> int -> char
val concat : string -> string list -> string concat sep ss concatenates the list of strings ss, inserting the separator string sep between each.
val starts_with : prefix:string -> string -> bool
val ends_with : suffix:string -> string -> bool
val contains : string -> char -> bool
val sub : string -> int -> int -> string
sub s pos len is a string of length len, containing the substring of s that starts at position pos and has length len.

Lists

val nth_opt : 'a list -> int -> 'a option
val concat : 'a list list -> 'a list
val filter_map : ('a -> 'b option) -> 'a list -> 'b list
filter_map f l applies f to every element of l, filters out the None elements and returns the list of the arguments of the Some elements.
val for_all : ('a -> bool) -> 'a list -> bool
for_all f [a1; ...; an] checks if all elements of the list satisfy the predicate f.
val for_all2 : ('a -> 'b -> bool) -> 'a list -> 'b list -> bool
val exists : ('a -> bool) -> 'a list -> bool
exists f [a1; ...; an] checks if at least one element of the list satisfies the predicate f.
val find : ('a -> bool) -> 'a list -> 'a
find f l returns the first element of the list l that satisfies the predicate f. Raises
val find_opt : ('a -> bool) -> 'a list -> 'a option
val sort : ('a -> 'a -> int) -> 'a list -> 'a list
val stable_sort : ('a -> 'a -> int) -> 'a list -> 'a list
Same as List.sort, but the sorting algorithm is guaranteed to be stable (i.e. elements that compare equal are kept in their original order).
val flatten : 'a list list -> 'a list concatenates all lists

Associative lists

val assoc : 'a -> ('a * 'b) list -> 'b
val assoc_opt : 'a -> ('a * 'b) list -> 'b option
val mem_assoc : 'a -> ('a * 'b) list -> bool
Same as List.assoc, but simply return true if a binding exists
val remove_assoc : 'a -> ('a * 'b) list -> ('a * 'b) list
remove_assoc a l returns the list of pairs l without the first pair with key a,
val split : ('a * 'b) list -> 'a list * 'b list
val combine : 'a list -> 'b list -> ('a * 'b) list