System Functions and their Types in Shen

- **absvector**
  _
  Given a non-negative integer returns a vector in the native platform.

- **absvector?**
  A → boolean
  Recognisor for native vectors.

- **address->**
  _
  Given an absolute vector A, a positive integer *i* and a value V places V in the A[*i*]th position.

- **<-address**
  _
  Given an absolute vector A, a positive integer *i* retrieves V from the A[*i*]th position.

- **adjoin**
  A → (list A) → (list A)
  Conses an object to a list if it is not already an element..

- **and**
  boolean → boolean → boolean
  Boolean and.

- **append**
  (list A) → (list A) → (list A)
  Appends two lists into one list.

- **arity**
  A → number
  Given a Shen function, returns its arity otherwise -1.

- **boolean?**
  A → boolean
  Recognisor for booleans.

- **bound?**
  symbol → boolean
  Returns true if the variable is globally bound.

- **cd**
  string → string
  Changes the home directory. (cd "Prog") causes (load "hello_world.txt") to load Prog/hello_world.txt. (cd "") is the default.

- **close**
  (stream A) → (list B)
  Closes a stream returning the empty list.

- **cn**
  string → string → string
  Concatenates two strings.

- **concat**
  –
  Concatenates two symbols or booleans.

- **cons**
  –
  A special form that takes an object *e* of type A and a list *l* of type (list A) and produces a list of type (list A) by adding *e* to the front of *l*.

- **cons?**
  A → boolean
  Returns true iff the input is a non-empty list.

- **declare**
  –
  Takes a function name *f* and a type *t* expressed as a list and gives *f* the type *t*.

- **define**
  –
  Top level form for Shen definitions.

- **defmacro**
  –
  Top level form for Shen macros.

- **defprolog**
  –
  Top level form for Shen Prolog definitions.
- **destroy**
  (A → B) → symbol
  Receives the name of a function and removes it and its type from the environment.

- **difference**
  (list A) → (list A) → (list A)
  Subtracts the elements of the second list from the first.

- **do**
  A → (B → B)
  Returns its last argument; polyadic courtesy of the reader.

- **element?**
  A → (list A) → boolean
  Returns true iff the first input is an element in the second.

- **empty?**
  A → boolean
  Returns true iff the input is [ ].

- **error**
  _
  A special form: takes a string followed by *n* (*n* ≥ 0) expressions. Prints error string.

- **error-to-string**
  exception → string
  Maps an error message to the corresponding string.

- **eval**
  _
  Evaluates the input.

- **eval-kl**
  _
  Evaluates the input as a Kλ expression.

- **explode**
  A → (list string)
  Explodes an object to a list of strings.

- **external**
  symbol → (list symbol)
  Given a package name, returns the list of symbols external to that package.

- **fix**
  (A → A) → (A → A)
  Applies a function to generate a fixpoint.

- **freeze**
  A → (lazy A)
  Returns a frozen version of its input.

- **fst**
  (A * B) → A
  Returns the first element of a tuple.

- **function**
  (A → B) → (A → B)
  Maps a symbol to the function which it denotes.

- **gensym**
  symbol → symbol
  Generates a fresh symbol or variable from a symbol.

- **get-time**
  symbol → number
  For the argument *run* or *real* returns a number representing the real or run time elapsed since the last call. One of these options must be supported. For the argument *unix* returns the Unix time.

- **get**
  –
  takes a symbol S, a pointer P and optionally a vector V and returns the value in V pointed by P from S (if one exists) or an error otherwise. If V is omitted the global property vector is used.

- **hash**
  A → number → number
  Returns a hashing of the first argument subject to the restriction that the encoding must not be greater than the second argument.

- **head**
  (list A) → A
  Returns the first element of a list; if the list is empty returns an error

- **hd**
  (list A) → A
  Returns the first element of a list; if the list is empty returns an unspecified object

- **hdstr**
  string → string
  Returns the first element of a string.

- **hdv**
  (vector A) → A
  Returns the first element of a standard vector.

- **if**
  boolean → A → A → A
  takes a boolean *b* and two expressions *x* and *y* and evaluates *x* if *b* evaluates to true and evaluates *y* if *b* evaluates to false.

- **implementation**
  → string
  Returns a string denoting the implementation on which Shen is running (SBCL etc).

- **include**
  (list symbol) → (list symbol)
  Includes the datatype theories or synonyms for use in type checking.

- **include-all-but**
  (list symbol) → (list symbol)
  Includes all loaded datatype theories and synonyms for use in type checking apart from those entered.

- **inferences**
  A → number
  The input is ignored. Returns the number of logical inferences executed since the last call to the top level.

- **input**
  _
  0-place function. Takes a user input *i* and returns the normal form of *i*.

- **input+**
  _
  Special form. Takes inputs of the form **: <expr>**. Where *d*(**<expr>**) is the type denoted by the choice of expression (e.g. 'number' denotes the type number). Takes a user input *i* and returns the normal form of *i* given *i* is of the type *d*(**<expr>**).

- **integer?**
  A → boolean
  Recognisor for integers.

- **intern**
  _
  Maps a string to a symbol.

- **intersection**
  (list A) → (list A) → (list A)
  Computes the intersection of two lists.

- **it**
  → string
  Returns the last input to standard input embedded in a string.

- **lambda**
  _
  Builds a lambda expression from a variable and an expression.

- **language**
  → string
  Returns a string denoting the language on which Shen is running.

- **length**
  (list A) → number
  Returns the number of elements in a list.

- **limit**
  (vector A) → number
  Returns the maximum index of a vector.

- **lineread**
  _
  Top level reader of read-evaluate-print loop. Reads elements into a list. **lineread** terminates with carriage return when brackets are balanced. ^ aborts lineread.

- **load**
  string → symbol
  Takes a file name and loads the file, returning **loaded** as a symbol.

- **macroexpand**
  _
  Expand an expression by the available macros.
- **map**
  (A → B) → (list A) → (list B)
  The first input is applied to each member of the second input and the results consed into one list.

- **mapcan**
  (A → (list B)) → (list A) → (list B)
  The first input is applied to each member of the second input and the results appended into one list.

- **make-string**
  _
  A special form: takes a string followed by *n* (*n* ≥ 0) well-typed expressions; assembles and returns a string.

- **maxinferences**
  number → number
  Returns the input and as a side-effect, sets a global variable to a number that limits the maximum number of inferences that can be expended on attempting to type check a program. The default is 10^6.

- **nl**
  number → number
  Prints *n* new lines.

- **not**
  boolean → boolean
  Boolean not.

- **nth**
  number → (list A)→ A
  Gets the nth element of a list numbered from 1.

- **number?**
  A → boolean
  Recognisor for numbers.

- **n->string**
  number → string
  Given a number *n* returns a unit string whose ASCII number is *n*.

- **occurrences**
  A → B → number
  Returns the number of times the first argument occurs in the second.

- **occurs-check**
  symbol → boolean
  Receives either + or - and enables/disables occur checking in Prolog, datatype definitions and rule closures.   The default is +.

- **open**
  _
  Takes two arguments; the location from which it is drawn and the direction (*in* or *out*) and creates either a source or a sink stream.

- **or**
  boolean → (boolean → boolean)
  Boolean or.

- **os**
  → string
  Returns a string denoting the operating system on which Shen is running.

- **output**
  _
  A special form: takes a string followed by *n* (*n* ≥ 0) well-typed expressions; prints a message to the screen and returns an object of type string (the string "done").

- **package**
  _
  Takes a symbol, a list of symbols and any number of expressions and places them in a package.

- **package-exists?**
  symbol → boolean
  Returns **true** if the symbol names a package else returns **false**.

- **pos**
  string → number → string
  Given a string and a natural number *n* returns the *n*th unit string numbering from zero.

- **pr**
  string → (stream out) → string
  Takes a string, a sink object and prints the string to the sink, returning the string as a result. If no stream is supplied defaults to the standard output.

- **preclude**
  (list symbol) → (list symbol)
  Removes the mentioned datatype theories and synonyms from use in type checking.

- **preclude-all-but**
  (list symbol) → (list symbol)
  Removes all the datatype theories and synonyms from use in type checking apart from the ones given.

- **print**
  A → A
  Takes an object and prints it, returning it as a result.

- **profile**
  (A → B) → (A → B)
  Takes a function represented by a function name and inserts profiling code returning the function as an output.

- **profile-results**
  (A → B) → ((A → B) * number)
  Takes a profiled function f and returns the total run time expended on f since profile-results was last invoked..

- **ps**
  _
  Receives a symbol denoting a Shen function and prints the Kλ source code associated with the function.

- **put**
  _
  3-place function that takes a symbol S, a pointer P (a string symbol or number), and an expression E. The pointer P is set to point from S to the normal form of E which is then returned.

- **read**
  (stream in) → unit
  Takes a stream and reads off the first Shen token; defaults with zero arguments to standard input.

- **read-byte**
  (stream in) → number
  Takes a source and reads the first byte off it; defaults with zero arguments to standard input.

- **read-file**
  string → (list unit)
  Returns the contents of an ASCII file designated by a string. Returns a list of units, where unit is an unspecified type.

- **read-file-as-bytelist**
  string → (list number)
  Returns the contents of an ASCII file designated by a string as a list of bytes.

- **read-file-as-string**
  string → string
  Returns the string contents of an ASCII file designated by a string.

- **read-from-string**
  string → (list unit)
  Reads a list of expressions from a string.

- **remove**
  A → (list A) → (list A)
  Removes all occurrences of an element from a list.

- **require**
  symbol → string → symbol → (list symbol)
  Takes the purported name of a package and a string and the argument weak or strong. If weak and the package does not exist, the file denoted by the string is loaded and the list of external symbols to the package is returned. If strong, the file is always loaded and the external symbols returned.

- **reverse**
  (list A) → (list A)
  Reverses a list.

- **simple-error**
  string → A
  Given a string, raises it as an error message.

- **snd**
  (A * B) → B
  Returns the second element of a tuple.

- **specialise**
  symbol → symbol
  Receives the name of a function and turns it into a special form. Special forms are not curried during evaluation or compilation.

- **spy**
  symbol → boolean
  Receives either + or – and respectively enables/disables tracing the operation of $\mathcal{T}$*.

- **step**
  symbol → boolean
  Receives either + or – and enables/disables stepping in the trace.

- **stinput**
  → (stream in)
  Returns the standard input stream.
- **stoutput**
  → (stream out)
  Returns the standard output stream.

- **str**
  A → string
  Given an atom (boolean, symbol, string, number) flanks it in quotes. For other inputs an error may be returned.

- **string?**
  A → boolean
  Recognisor for strings.

- **string->n**
  string → number
  Maps a unti string to its code point.

- **subst**
  –
  Given (*subst x y z*) replaces *y* by *x* in *z* where *z* is a list or an atom.

- **sum**
  (list number) → number
  Sums a list of numbers.

- **symbol?**
  A → boolean
  Recognisor for symbols.

- **systemf**
  symbol → (list symbol)
  Gives the symbol the status of an identifier for a system function; its definition may not be overwritten. Returns the list of symbols with this status.

- **tail**
  (list A) → (list A)
  Returns all but the first element of a non-empty list.

- **tc**
  symbol → boolean
  Receives either + or – and respectively enables/disables static typing.

- **tc?**
  A → boolean
  Returns true iff typechecking is enabled.

- **thaw**
  (lazy A) → A
  Receives a frozen input and evaluates it to get the unthawed result..

- **time**
  _
  Prints the run time for the evaluation of its input and returns its normal form.

- **tl**
  _
  Returns the tail of a list; for [] the result is platform dependent.

- **tlstr**
  string → string
  Returns the tail of a string.

- **tlv**
  (vector A) → (vector A)
  Returns the tail of a non-empty vector.

- **track**
  symbol → symbol
  Tracks the I/O behaviour of a function.

- **trap-error**
  A → (exception → A) → A
  Tracks the I/O behaviour of a function.

- **tuple?**
  A → boolean
  Recognisor for tuples.

- **type**
  _
  Used under type checking; takes an expression *e* and a type A; *e* is evaluated only if *e* inhabits A.

- **undefmacro**
  symbol → symbol
  Removes a macro.

- **union**
  (list A) → (list A) → (list A)
  Forms the union of two lists.

• **unprofile**
(A → B) → (A → B)
Unprofiles a function.

• **unspecialise**
symbol → symbol
Receives the name of a function and deletes its special form status.

• **untrack**
symbol → symbol
Untracks a function.

• **value**
_
Applied to a symbol, returns the global value assigned to it.

• **variable?**
A → boolean
Applied to a variable, returns true.

• **version**
string → string
Changes the version string displayed on startup.

• **vector**
number → (vector A)
Creates a vector of size *n*.

• **vector?**
A → boolean
Recognises a standard vector.

• **vector->**
(vector A) → number → A → (vector A)
Given a vector V and an index *i* and object *o*, assigns *o* to V[*i*].

• **<-vector**
(vector A) → number → A
Given a vector V and an index *i* and object *o*, assigns *o* to V[*i*].

• **vector?**
A → boolean
Recognisor for standard vectors.

• **write-byte**
number → (stream out) → number
Takes a byte as an integer *n* between 0 and 255 and writes the corresponding
byte to the stream returning *n*.

- **write-to-file**
  string → A → A
  Writes the second input into a file named in the first input. If the file does not exist, it is created, else it is overwritten. If the second input is a string then it is written to the file without the enclosing quotes. The second input is returned.

- **y-or-n?**
  string → boolean
  Prints the string as a question and returns true for y and false for n.

- **@p**
  _
  Takes *n* (*n* > 1) inputs and forms the tuple.

- **@s**
  _
  Takes *n* (*n* > 1) strings and forms their concatenation

- **@v**
  _
  Takes *n* inputs, the last being a vector V and forms a vector of these elements appended to the front of V.

- **$**
  _
  Used by the reader; the argument is read in as an exploded list of unit strings.

.

- +
  number → number → number
  Number addition.

- –
  number → number → number
  Number subtraction.

- *
  number → number → number
  Number multiplication.

- /
  number → number → number
  Number division.

- /.
  _

Abstraction builder, receives *n* variables and an expression; does the job of a (nested) λ in the lambda calculus.

- >
  number → number → boolean
  Greater than.

- <
  number → number → boolean
  Less than.

- =
  A → A → boolean
  Equal to.

- ==
  A → B → boolean
  Equal to.

- >=
  number → number → boolean
  Greater than or equal to.

- <=
  number → number → boolean
  Less than or equal to.