# Unstable-Languages-beta: SIMPLE-THR-Funcons-Index

The PLanCompS Project

`Unstable-Languages-beta/SIMPLE-THR/SIMPLE-THR-Funcons-Index/SIMPLE-THR-Funcons`

## Computations

### Normal computation

#### Flowing

[ *Funcon* sequential
*Alias* seq
*Funcon* effect
*Funcon* if-true-else
*Alias* if-else
*Funcon* while-true
*Alias* while
*Funcon* atomic ]

#### Giving

[ *Entity* given-value
*Funcon* initialise-giving
*Funcon* give
*Funcon* given
*Funcon* no-given
*Funcon* left-to-right-repeat
*Funcon* left-to-right-filter ]

*Suggestions for improvement: `plancomps@gmail.com`.
Issues: `https://github.com/plancomps/CBS-beta/issues`.

### Binding

[ *Type* environments
*Alias* envs
*Datatype* identifiers
*Alias* ids
*Funcon* initialise-binding
*Funcon* bind-value
*Alias* bind
*Funcon* bound-value
*Alias* bound
*Funcon* scope
*Funcon* collateral ]

### Generating

[ *Type* atoms
*Funcon* fresh-atom ]

### Storing

[ *Entity* store
*Funcon* initialise-storing
*Datatype* variables
*Alias* vars
*Funcon* allocate-variable
*Alias* alloc
*Funcon* allocate-initialised-variable
*Alias* alloc-init
*Funcon* assign
*Funcon* assigned
*Funcon* un-assign ]

### Interacting

### Input

[ *Funcon* read ]

### Output

[ *Funcon* print ]

## Abnormal computation

### Terminating abruptly

[ *Entity* abrupted
*Funcon* handle-abrupt ]

### Failing

[ *Funcon* finalise-failing
*Funcon* fail
*Funcon* else
*Funcon* checked
*Funcon* check-true ]

### Throwing

[ *Funcon* finalise-throwing
*Funcon* throw
*Funcon* handle-thrown ]

### Returning

[ *Funcon* return
*Funcon* handle-return ]

# Values

## Value Types

[ *Type* values
*Alias* vals
*Funcon* is-value
*Alias* is-val
*Funcon* when-true
*Alias* when
*Type* ground-values
*Alias* ground-vals
*Funcon* is-equal
*Alias* is-eq ]

## Primitive values

### Booleans

[ *Datatype* booleans
*Alias* bools
*Funcon* true
*Funcon* false
*Funcon* not
*Funcon* and ]

### Integers

[ *Type* positive-integers
*Alias* pos-ints
*Type* natural-numbers
*Alias* nats
*Funcon* natural-successor
*Alias* nat-succ
*Funcon* natural-predecessor
*Alias* nat-pred
*Funcon* integer-add
*Alias* int-add
*Funcon* integer-subtract
*Alias* int-sub
*Funcon* integer-multiply
*Alias* int-mul
*Funcon* integer-divide
*Alias* int-div
*Funcon* integer-modulo
*Alias* int-mod
*Funcon* integer-negate
*Alias* int-neg
*Funcon* integer-is-less
*Alias* is-less
*Funcon* integer-is-less-or-equal
*Alias* is-less-or-equal
*Funcon* integer-is-greater
*Alias* is-greater
*Funcon* integer-is-greater-or-equal
*Alias* is-greater-or-equal
*Funcon* decimal-natural
*Alias* decimal ]

### The null value

[ *Datatype* null-type
*Funcon* null-value
*Alias* null ]

## Composite values

### Sequences of values

[ *Funcon* length
*Funcon* index
*Funcon* first
*Funcon* second ]

### Tuples

[ *Datatype* tuples
*Funcon* tuple-elements ]

### Lists

[ *Datatype* lists
*Funcon* list-elements
*Funcon* list-cons
*Alias* cons
*Funcon* list-head
*Alias* head
*Funcon* list-tail
*Alias* tail
*Funcon* list-append ]

### Vectors

[ *Datatype* vectors
*Funcon* vector
*Funcon* vector-elements ]

### Sets

[ *Type* sets
*Funcon* is-in-set
*Funcon* is-subset
*Funcon* set-unite
*Funcon* set-intersect
*Funcon* set-difference
*Funcon* some-element ]

### Maps

[ *Type* maps
*Funcon* map
*Funcon* map-lookup
*Alias* lookup
*Funcon* map-domain
*Alias* dom
*Funcon* map-override
*Funcon* map-unite
*Funcon* map-delete ]

## Abstraction values

### Generic abstractions

[ *Type* abstractions
*Funcon* abstraction
*Funcon* closure ]

### Thunks

[ *Datatype* thunks
*Funcon* thunk ]

### Functions

[ *Datatype* functions
*Funcon* function
*Funcon* apply
*Funcon* supply ]

### Patterns

[ *Datatype* patterns
*Funcon* pattern
*Funcon* match ]