|  |  |
| --- | --- |
| TEA PRIMITIVE | SEMANTICS |
| V: | |  |  | | --- | --- | | NAME | Vault | | PURPOSE | Store and enable operating on stored things | | SYNTAX  & SEMANTICS | |  | | --- | | v: | | Store AI into the default vault (unnamed vault) – stores the EMPTY STRING (TEA Null/None value) if AI is not yet set. Returns AI | | v:vNAME  v:vNAME:vVALUE | | With only vNAME specified, store AI in a vault with that name. With vVALUE specified, store that instead of AI. Returns AI | | v!: | | Return the length of what is stored in the default vault. | | v!:STR | | Return the length of string STR | | v\*:vNAME  v\*:vNAME:vVALUE | | Store value vVALUE in vault vNAME overriding where necessary. Returns AI | | v\*!: vNAME | | Return the length of what is stored in the vault vNAME. Without vNAME, is like v!: | |  | | NOTES | Every useful programming language needs some simple way to not only store or perhaps temporarily hold many bits of data, but also, be able to access or reference them by name, as well as be able to operate on data at rest. In TEA, the V-command space makes this possible, and it is the closest facility in TEA, to what other languages offer as variables. Because most TEA programs will be operating on AI, the v: and v!: primitives offer a means to easily store or tell the length of AI without explicitly referencing it by name.  The v!: and v\*!: primitives offer the only straightforward way to determine the size of things in TEA – as all things being processed in TEA are strings, it comes in very handy for controlling certain operations based on size or counts.  The most basic example for how to return the length of any input is the following simple TEA program:  i!:ABC|v:|v!:  It should return just “3” for this example string “ABC”.  For more involved applications, see the following program that will help generate an 8-character password from any given non-empty initial string:  I!:{SOMEVALUE}  G!:|A!: # glue and anagrammatize seed  V:vSRC|V\*!:vSRC # store it in vault and return length  Q:0 # quit if seed was empty  V:vPWD:{} #init password with empty string  L:lTEST  V\*!:vPWD # get current length of password  F!:8:lPRODUCE:lRETURN # produce if not yet 8-chars long otherwise return  L:lPRODUCE  Y:vSRC #fetch the seed  A!:|D!:^. # shuffle seed and pick only first character  V:vCHOSEN # store picked letter in a vault  G\*:{}:vPWD:vCHOSEN # append it to current password  V:vPWD | V\*!:vPWD # override password and return its length  J:lTEST # return to the password length test  L:lRETURN  Y:vPWD # return final password  This non-trivial TEA program shall generate passwords like “LEEOSVVM”, “SSAMLUVE”, “UAEEMOLL” etc. |  |  | |