|  |  |
| --- | --- |
| TEA PRIMITIVE | SEMANTICS |
| Y: | |  |  | | --- | --- | | NAME | Yank | | PURPOSE | Return things from memory | | SYNTAX  & SEMANTICS | |  | | --- | | y: | | Return whatever is currently stored in the default vault.  (the unnamed vault) | | y:vNAME | | Return whatever is stored in vault with name vNAME | | y!: | | Return the length of whatever is currently stored in the default vault.  (the unnamed vault) | | y!:vNAME | | Return whatever is currently stored in vault vNAME | | y\*: | | Return whatever was the external, user provided initial input to the TEA program – the initial AI | | y\*:vNAME | | Return whatever is currently stored in vault vNAME | | y\*!:vNAME | | Return the length of whatever is currently stored in vault vNAME. | | y\*!: | | Return the length of whatever was the external, user provided initial input to the TEA program – the initial AI | |  | | NOTES | Yanking is essentially the opposite operation of vaulting in TEA. The Y-command also allows TEA programs to correctly reference original user-provided input in a TEA program – to help avoid the idiomatic inclusion of default inputs as part of common practice in TEA programming---such as with the use of i!: and w!: commands, from making access to original user-provided input impossible when needed.  The Y-Inverse commands, like V!, also allow for a straightforward way to determine the size of strings in TEA – for Y, only possible with strings already held inside some vault, while v!:STR also can help determine the length of explicit, inline string values.  A useful test of the Y: facility is the following program that should greet one with “Hello X” if one responds to the prompt with the name “X”:  i!:What is your name?|i:|v:vNAME|v:vGREET:{Hello }|v:vG:{}|g\*::vGREET:vNAME|v:vGRT|i:TEST|y:vGRT |  |  | |