Documentation

Junior R. Ribeiro

September 13, 2019

List of Tables

1 Class __workspace__ 1
2 Class Matrix 2

Table 1: Class __workspace__

Class: __workspace__	
	(empty) ..... //there is no global constants on workspace.
+	warn(msg,endline*) This function prints "WARN: msg" in red on terminal. » msg is a string; » endline is a boolean.
+	info(msg,endline*) This function prints "INFO: msg" in blue on terminal. » msg is a string; » endline is a boolean.
+	rcout(msg,endline*) This function prints "msg" in red on terminal. » msg is a string; » endline is a boolean.
+	bcout(msg,endline*) This function prints "msg" in blue on terminal. » msg is a string; » endline is a boolean.
« Legend »	
+	public
*	optional

Class __workspace__ ends here.

Table 2: Class `Matrix`

Class: `Matrix`	
−	`.me` ..... // Pointer to pointer like **me (the Matrix itself).
−	`.isdestroyed` ..... // Boolean indicating whether the object was destroyed.
−	`.m` ..... // The number of rows of the Matrix. A positive integer.
−	`.n` ..... // The number of columns of the Matrix. A positive integer.
+	`Matrix(m,n)` ..... // The constructor method.
+	`.throwisdestroyed(functionName)` This function raises an error and exits the program always when it is attempted to use a destroyed Matrix. » functionName is a string indicating the name of what function is attempting to use the Matrix.
+	`.set(i,j,value)` ...
+	`.get(i,j)` ...
+	`.sum(otherMatrix)` ...
+	`.sub(otherMatrix)` ...
+	`.mul(otherMatrix)` ...
+	`.fromuser(clearPrompt)` ...
+	`.print( )` ...
+	`.shape( )` ...
+	`.shape1( )` ...
+	`.shape2( )` ...
+	`.destroy( )`

(continued on next page)

Table 2 – Class `Matrix` *(continued from previous page)*

	This method desallocates the matrix and frees the memory. The integers m and n still remain on memory.
« Legend »	
+	`public`
–	`private`

Class `Matrix` ends here.