Indice delle **Funzioni** esposte dalla libreria **myMatrix**

Per richiamare la libreria inserire il codice seguente:
from myMatrix import *
objM = myMatrix()

Metodo(arg1, arg2, …)	**scopo**	**Esempi chiamata con …**	**… valore ritornato**
dim (M) MatrixSize (M)	fornisce le dimensioni della matrice	M = [[2,-1,0],[3,6,-1],[1,2,1]] objM.dim (M)	[3,3] # è una lista
column (M, num_col)	restituisce una colonna di M; n_col parte da 0	M = [[2,-1,0],[3,6,-1],[1,2,1]] objM.column (M,2)	[0,-1,-1] # è una lista
row (M, num_row)	restituisce una riga di M; n_row parte da 0	M = [[2,-1,0],[3,6,-1],[1,2,1]] objM.raw (M,2)	[1,2,1] # è una lista
printMatrix (M)	stampa la matrice in formato righe-colonne	M = [[2,-1,0],[3,6,-1],[1,2,1]] objM.printMatrix(M)	[2,-1,0 3,6,-1 1,2,1]
maxDigitLen	Trova l'elemento più lungo, per dimensionare la griglia di stampa	Viene utilizzata internamente a printMatrix	
MatrixCreate (n_rows, n_cols, opz: val)	Crea una matrice di tipo n_rows X n_cols, eventualmente riempita con val	objM.MatrixCreate (3,2,val=9)	[9, 9 9,9 9, 9]
MatrixIsEmpty (M)	Stabilisce se M è vuota cioè non ha elementi	M=None / M=[6,2,7] objM.MatrixIsEmpty (M)	true / false
IsVectorAsMatrix (M)	Stabilisce se M è un vettore	M=[[6,2,7]] / M=[6,2,7] objM.IsVectorAsMatrix (M)	true / false
MatrixIdentity (dim)	Crea una matrice identità di tipo dimXdim	objM.MatrixIdentity(2)	[1,0 0,1]
MatrixZero (dim)	Crea una matrice nulla (con tutti zero) di tipo dimXdim	objM.MatrixZero(2)	[0,0 0,0]
MatrixTranspose (M)	Fornisce la trasposta di M	M = [[2,-1,0],[3,6,-1],[1,2,1]] objM.MatrixTranspose(M)	[2,3,1 1,6,2 0,-1,1]
MatrixMult2 (M1, M2)	Moltiplica 2 matrici	M1 = [[2,-1,0],[3,6,-1],[1,2,1]]	[-1.00 3.00 6.00

		M2 = [[1,2,4],[3,1,2],[6,0,3]] objM.MatrixMult2(M1, M2)	15.00 12.00 21.00 13.00 4.00 11.00]
MatrixSum2 (M1, M2)	Somma 2 matrici	M1 = [[2,-1,0],[3,6,-1],[1,2,1]] M2 = [[1,2,4],[3,1,2],[6,0,3]] objM.MatrixSum2(M1, M2)	[3.00 1.00 4.00 6.00 7.00 1.00 7.00 2.00 4.00]
MatrixDiff2 (M1, M2)	Sottrae 2 matrici: M1 – M2	M1 = [[2,-1,0],[3,6,-1],[1,2,1]] M2 = [[1,2,4],[3,1,2],[6,0,3]] objM.MatrixDiff2(M1, M2)	[1.00 -3.00 -4.00 0.00 5.00 -3.00 5.00 2.00 -2.00]
dot (vettore1,vettore2)	Esegue il prodotto scalare di due vettori	V1 = [3,9] V2 = [2,1] objM.dot(V1, V2)	15
length(vettore) norm (vettore)	Fornisce il modulo di un vettore	V = [2,1] objM.length (V1)	2.23606
ScalarMult (scalare, M)	Esegue il prodotto di M per uno scalare	s = 2 M = [[2,-1,0],[3,6,-1],[1,2,1]] objM.ScalarMult(s, M)	[4.00 -2.00 0.00 6.00 12.00 -2.00 2.00 4.00 2.00]
MatrixInvert (M)	Fornisce la matrice inversa di M	M = [[2,-1,0],[3,6,-1],[1,2,1]] objM.MatrixInvert(M)	[0.40 0.05 0.05 0.20 0.10 0.10 0.00 -0.25 0.75]
MatrixRotX (angle)	Fornisce la matrice della rotazione di angle° attorno X	angolo = 90 objM. MatrixRotX (angolo)	[1.00 0.00 0.00 0.00 0.00 -1.00 0.00 1.00 0.00]
MatrixRotY (angle)	Fornisce la matrice della rotazione di angle° attorno Y	angolo = 90 objM. MatrixRotY (angolo)	[0.00 0.00 1.00 0.00 1.00 0.00 1.00 0.00 0.00]
MatrixRotZ (angle)	Fornisce la matrice della rotazione di angle° attorno Z	objM. MatrixRotZ (angolo)	[0.00 -1.00 0.00 1.00 0.00 0.00 0.00 0.00 1.00]
MatrixRotV(angle, vector)	Fornisce la matrice della rotazione di angle° attorno all'asse definito da vector	angolo = 90 asse_rot = [2,1,0] objM.MatrixRotV (angolo, asse_rot)	[0.80 0.40 0.45 0.40 0.20 -0.89 0.45 0.89 0.00]