# Quantum Algebra (QuaAlg) library an assistant to algebra calculations

## General:

The program library QuaAlg is a module for python to assist you in your calculation inside of linear algebra. It is made especially to assist physics in quantum mechanics but is made in such a way to reach a more broad crowed of mathematicians, software engineers and physicst.

## Structure of Matrix and vectors:

It is coded without a class for matrix and vector to assist you in your coding more generally. This allows you to use matrix calculation and vector calculation more broadly in all your programs. If you stick to the module's notation. The notation is that a vector is a list of numbers, Int or float. And matrix is a list of lists. Where the first list is a refence to which row of element in the matrix we are in and the second is the elements of that row.

## Structure and functions in QuaAlg the basics:

**Quantum Algebra is a module to support math and quantum calculation**

Matrix form defined by M[i][j]
An example is
M= [[3,5,7],
[3,8,9],
[1,6,8]]
postions with value 5 is i=0 and j=1 postion of 9 is i=1 and j=2

If you want a complex matrix. Define K=[R,C]
Where R is the real part and C is the complex part
R and C is in the same form as the ordinary matrix

```
def scal_vec(v,s):#Multiplication between vector and scalor

return new_vector

def scal_mat(M,s):#Multiplication between matrix and scalor

return new_matrix

def def scalar_prod(v_1,v_2):#This is a function to take two vectors and return the scalar product

return result

def matrix_prod(M_1,M_2):#Multplies two matrix and create M_N

return M_N

def matrix_add(M_1,M_2):# Add two matrix to eachother return M
```

return M

def matrix_prod_possible(M_1,M_2):#Check if matrix size of M_1 and M_2 are of right size to multiply return if possible

return True/False

def column_vector(M,c):#Return a column vector in a matrix.

return [M[i][c],] for all i

def commute_matrix(M_1,M_2):#Check if matrix A, B commute or not.

return commuter

def check_matrix(M):#Check if matrix is properly made. No other variable then int or float in space. Return true or false

return True/False

def mat_sym(M): #Return if the matrix is symmetric or not

return True/False

def scal_com_mat(K):#Multplication between an complex matrix and a scalor calls it M_N

return M_N

def mat_her(K):# Return if the matrix is hermitaion or not

return True/False

def spin_mat(ro,theta):# Return the spin matrix to measure a quantum spin state

return [R,C]#R is the real part C is the complex part of the matrix