```
from sympy import *
from IPython.display import *
from sympy.polys.orderings import monomial_key
init_printing()

var(’a:z’)
var(’Gamma’)

def XVAR(vr,length):
    xa=str(vr)
    for i in range(length):
        xa+=str(i)+","+str(vr)
    xa+=str(length)
    return list(var(xa))

def SymPower(A,N):

    d=A.shape[0]
    X=XVAR("x",d-1);Y=XVAR("y",d-1)

    X.reverse()
    IT=sorted(itermonomials(X,N),key=monomial_key(’grevlex’, X))
    nd=binomial(N+d-1,N)
    L=IT[-nd:]
    X.reverse()

    Y.reverse()
    IT=sorted(itermonomials(Y,N),key=monomial_key(’grevlex’, Y))
    nd=binomial(N+d-1,N)
    LL=IT[-nd:]
    Y.reverse()

    XV=Matrix(X)
    B=A*XV

    nd=len(L)
    M=[]

    for i in range(nd):
        F=LL[i]
        for j in range(d):
            F=F.subs(Y[j],B[j])
        G=expand(F)
        M.append([G.coeff(L[k]) for k in range(nd)])

    MX=Matrix(1,nd,M[0])
    for i in range(1,nd):
        XM=Matrix(1,nd,M[i])
        MX=MX.col_join(XM)

    return MX

def SymmTraces(A,order):

    IX=eye(A.shape[0])
    delta=series(((IX-t*A).det())**(-1),t,0,order)
    return [delta.coeff(t,i) for i in range(order)]

def PowerTraces(A,order):
    IX=eye(A.shape[0])
    delta=series(trace((IX-t*A).inv()),t,0,order)
    return [delta.coeff(t,i) for i in range(order)]

def GAM(A,N):
        d=A.shape[0]
        S=SymPower(eye(d)+t*A,N)
        return S.diff(t).subs(t,0)

print(‘‘This provides macros for symmetric tensor powers.\n")
print(‘‘SymPower(A,N): a matrix A and degree N")
print(‘‘SymmTraces(A,n): matrix A and order of the series")
print(‘‘PowerTraces(A,n): matrix A and order of the series")
print(‘‘GAM(A,N): Lie map for matrix A in degree N")
```