In [84]:
```
n=6
g=(n-1)*(n-2)/2;
K=CyclotomicField(2*n)
z=K.gen()
zn=z^2
PP.<T>=PolynomialRing(K)
```

In [85]:
```
def IMod(a,b):
    ypol=a-b*floor(a/b)
    return ypol
def rat(x):
    return numerator(x)/denominator(x)
```

In [86]:
```
def polynom(V,Q,ka,la,nu,rho):
    if not(3.divides(n)) and nu > 0:
        print("3 does not divide",n, " and nu=",nu)
        return "ERROR"
        print("XXXXXXXXXX",V,Q,ka,la,nu,rho)
    if Q == 0:
        ex=IMod( nu*n/3,n)
        if V == 1:
            if rho == "triv" and nu==0:
                poly=1
            if rho == "sgn"  and nu==0:
                poly=T^n
            if rho == "triv" and nu==1:
                poly= T^(4*n/3)
            if rho == "sgn"  and nu==1:
                poly=T^(n/3)
            if rho == "triv" and nu==2:
                poly= T^(2*n/3)
            if rho == "sgn"  and nu==2:
                poly=T^(5*n/3)
            # print("--",poly)
            return poly

        if V == 2:
            poly=T^(nu*n/3)+T^(n+nu*n/3)
            # print(poly)
            return poly


        if V == 3:
            if rho == "triv":
                poly=T^ka+T^(n+ka)+T^(IMod(-2*ka,2*n))
            if rho == "sgn":
                # if ka <=floor(n/2):
                #     poly=T^ka+T^(n+ka)+T^(n-2*ka)
                # if ka >=ceil(n/2):
                #     poly=T^ka+T^(n+ka)+T^(3*n-2*k)
                poly=T^ka+T^(ka+n)+T^(IMod(-2*ka+n,2*n))
            return poly

        if V == 6:
            ee=IMod(-(ka+la),n)
            poly=T^ka+T^la+T^(n+ka)+T^(n+la)+T^ee+T^(ee+n)
            # if ka+la <=floor(n/2):
            #     poly=T^ka+T^la+T^(n+ka)+T^(n+la)+T^(2*n-(ka+la))+T^(n-(ka+
            # if ka+la >=ceil(n/2):
            #     poly=T^ka+T^la+T^(n+ka)+T^(n+la)+T^(2*n-(ka+la))+T^(3*n-(k
            return poly
```

```
    if Q == 1:
        if V == 1 and rho == "sgn":
            poly=T
            return poly
        if V == 1 and rho == "triv":
            poly=1
            return poly

        if V == 2:
            poly=1+T
            return poly

        if V == 3 and rho == "sgn":
            poly=1+2*T
            return poly
        if V == 3 and rho == "triv":
            poly=2+T
            return poly

        if V == 6:
            poly=3+3*T

    if Q == 2:

            pp=T^IMod(nu,3)

            if V == 1:
                poly=pp
                # poly=T^3

            if V == 2:
                poly=1+T+T^2 -pp

            if V == 3:
                poly=1+T+T^2
                # poly=T^3+T+T^2

            if V == 6:
                poly=2*(1+T+T^2)
                # poly=2*(T^3+T+T^2)


    return poly
```

In [91]:

```
def Hgen(V,ka,la,nu,rho):
    pol0=polynom(V,0,ka,la,nu,rho)
    pol1=polynom(V,1,ka,la,nu,rho)
    pol2=polynom(V,2,ka,la,nu,rho)
    # print("V=",V,"ka=",ka,"la=",la,"nu=",nu,"rho=",rho)
    # print("   pol0=",pol0)
    # print("   pol1=",pol1)
    # print("   pol2=",pol2)
    Dpol0=derivative(pol0,T)
    Dpol1=derivative(pol1,T)
    Dpol2=derivative(pol2,T)

    sumD=Dpol0(T=1)/(2*n) + Dpol1(T=1)/2 +  Dpol2(T=1)/3
    # print("             sumD=",sumD)

    sum1=pol0/(1-T^(2*n))+pol1/(1-T^2)+pol2/(1-T^3)

    # print(sum)
    H=V/(1-T)^2 -sumD/(1-T)-T*sum1/(1-T)
```

```
    H=numerator(H)/denominator(H)
    return H
```

In [161…

```
# Costructing the indices of a basis of holomorphic 1-differentials
L=[]
for i in range(n-3+1):
    for j in range(n-3-i+1):
        L.append([i,j])
Ls=[]
for el in L:
    Ls.append([n-3-el[0]-el[1],el[0]])
# create the matrix for the action on s
M=matrix(g)
for j in range(g):
    for i in range(g):
        if L[j] == Ls[i]:
            # print(i,j)
            M[i,j]=1
Lts=[]
for el in L:
    Lts.append([el[1],el[0]])
N=matrix(g)
for j in range(g):
    for i in range(g):
        if L[j] == Lts[i]:
            # print(i,j)
            N[i,j]=1

M=M.transpose()
N=N.transpose()
#-----acting on bases so need to take transpose, it makes a difference on mu

# S3=[[M,"s"],[M^2,"s^2"],[M^3,"id"],[N,"ts"],[M*N,"st"],[N*M,"t"]]

# S3=[[M,"s"],[M^2,"s^2"],[M^3,"id"],[N,"ts"],[M*N,"st"],[N*M,"t"]]
# S3=[[M,"s^2"],[M^2,"s"],[M^3,"id"],[N,"ts"],[M*N,"st"],[N*M,"t"]]
S3=[[M,"s"],[M^2,"s^2"],[M^3,"id"],[N,"ts"],[M*N,"t"],[N*M,"st"]] # <---


Dlist=[]
for k in range(n):
    for l in range(n):
        C=matrix(K,g)
        for i in range(g):
            el=L[i]
            C[i,i]=zn^(k*(el[0]+1)+l*(el[1]+1))
        C=C.transpose()
        Dlist.append([k,l,C])
        # print(C)


G=[]
for i in Dlist:
    for s in S3:
        # print(i[2]*s)
        # print("------------")
        G.append([s[0]*i[2],i[0],i[1],s[1]])
        # In first position is the matrix, then k,l,string determining the e

def chara(gg,V,ka,la,nu,rho):
    if rho=="triv":
        if V==1:
            ch=zn^(nu*n/3*(gg[1]+gg[2]))
```

```
            return ch
        if V==3:
            if gg[3]=="id":
                ch=zn^(ka*(gg[1]+gg[2]))+zn^(ka*(gg[1]-2*gg[2]))+zn^(ka*(gg[
                return ch
            if gg[3]=="ts":
                ch=zn^(ka*(gg[1]+gg[2]))
                return ch
            if gg[3]=="st":
                ch=zn^(ka*(gg[2]-2*gg[1]))
                return ch
            if gg[3]=="t":
                ch=zn^(ka*(gg[1]-2*gg[2]))
                return ch
            if gg[3]=="s" or gg[3]=="s^2":
                ch=0
                return ch
        if V==6:
            if gg[3]=="id":
                ch=zn^(ka*gg[1]+la*gg[2])+zn^(-(ka+la)*gg[1]+ka*gg[2])+zn^(l
                return ch
            else:
                return 0
    if rho=="sgn":
        if V==1:
            # print("gg[3]=",gg[3])
            if gg[3]=="id" or gg[3]=="s" or gg[3]=="s^2":
                ch=zn^(nu*n/3*(gg[1]+gg[2]))
            if gg[3]=="t" or gg[3]=="ts" or gg[3]=="st":
                ch=-zn^(nu*n/3*(gg[1]+gg[2]))
            return ch
        if V==3:
            if gg[3]=="id":
                ch=zn^(ka*(gg[1]+gg[2]))+zn^(ka*(gg[1]-2*gg[2]))+zn^(ka*(gg[
                return ch
            if gg[3]=="ts":
                ch=-zn^(ka*(gg[1]+gg[2]))
                return ch
            if gg[3]=="st":
                ch=-zn^(ka*(gg[2]-2*gg[1]))
                return ch
            if gg[3]=="t":
                ch=-zn^(ka*(gg[1]-2*gg[2]))
                return ch
            if gg[3]=="s" or gg[3]=="s^2":
                ch=0
                return ch
    if rho=="stan":
        if gg[3]=="id":
                cc=2
        if gg[3]=="ts":
                cc=0
        if gg[3]=="t":
                cc=0
        if gg[3]=="st":
                cc=0
        if gg[3]=="s" or gg[3]=="s^2":
                cc=-1
        ch=zn^(nu*n/3*(gg[1]+gg[2]))*cc
        return ch

def MolienF(V,ka,la,nu,rho):
    sum=0
    I=matrix.identity(K,g)
```

```
    for gg in G:
        de=(I-T*gg[0]).determinant()
        sum=sum+chara(gg,V,ka,la,nu,rho).conjugate()/de
        # sum=sum+chara(gg,V,ka,la,nu,rho)/de
        # detList.append(de)
    F=1/(6*n^2)*sum
    # print(F)
    # print("-------")
    # print(SR(F).series(T))
    return [F,SR(F).series(T)]
```

In [141…

```
def reverse(nu,rho):
    if nu !=0 or V==3:
        if rho== "triv":
            rr="sgn"
        if rho== "sgn":
            rr="triv"
        if rho == "stan":
            rr="stan"
    else:
        rr=rho
    return rr
```

In [162…

```
repList=[
    [1,0,0,0,"triv"],
    [1,0,0,1,"triv"],
    [1,0,0,2,"triv"],
    [1,0,0,0,"sgn"],
    [1,0,0,1,"sgn"],
    [1,0,0,2,"sgn"],
    [2,0,0,0,"stan"],
    [2,0,0,1,"stan"],
    [2,0,0,2,"stan"],
    [3,3,3,0,"triv"],
    [3,1,1,0,"triv"],
    [3,5,5,0,"triv"],
    [3,3,3,0,"sgn"],
    [3,1,1,0,"sgn"],
    [3,5,5,0,"sgn"],
    [6,0,1,0,"triv"],
    [6,0,2,0,"triv"],
    [6,1,2,0,"triv"],
    [6,3,4,0,"triv"]]
sum=0
Msum=0
for el in repList:
        V=el[0]
        ka=el[1]
        la=el[2]
        nu=el[3]
        rho=el[4]
        HH=Hgen(V,ka,la,nu,rho)
        # print("V=",V,"ka=",ka,"la=",la,"nu=",nu,"rho=",rho)
        # print("  HH=  ",SR(HH).series(T))

        HHor=HH


        rr=reverse(nu,rho)

        Moli=MolienF(V,-ka,-la,-nu,rr)

        MM=Moli[1].coefficients()[1][0]
```

```
        # MM0=MolienF(V,ka,la,nu,rho)[1].coefficients()[1][0]
        if el==[1,0,0,0,"triv"]:
            HH=HH+T
        HHc=rat(HH+MM)

        if V == 1 or V==2:
            kk=n/3*nu
            ll=kk
        else:
            kk=ka
            ll=la

        # #------Uncoment for Latex output-----------
        # # print(latex(rat(HHor)), " & ", latex(rat(HH)), " & ", latex(rat(
        # print( V , " & ", kk , " & ", ll, " & RRho_{\mathrm{", rho , "}} &
        # # print("\hline")
        # #----------------------------------------

        #-----------Uncomment for screen printing
        print("V=",V,"ka=",ka,"la=",la,"nu=",nu,"rho=",rho)
        print(" HH=  ",HHor)
        print("MMdual=",MM)
        print("HHc=",HHc)

        print("-----------------------------------------------------------------
        #-----------

        sum=sum+V*(HHc)
```

```
V= 1 ka= 0 la= 0 nu= 0 rho= triv
 HH=   (T^8 - T^7 + T^6 + T^2 - 2*T + 1)/(T^13 - T^12 - T + 1)
MMdual= 0
HHc= (T^14 - T^13 + T^8 - T^7 + T^6 - T + 1)/(T^13 - T^12 - T + 1)
--------------------------------------------------------------------------------
V= 1 ka= 0 la= 0 nu= 1 rho= triv
 HH=   (T^12 - T^11 + T^10 - T^9 + T^6 - T^5 + T^4)/((T^12 - 1)*(T - 1))
MMdual= 0
HHc= (T^12 - T^11 + T^10 - T^9 + T^6 - T^5 + T^4)/((T^12 - 1)*(T - 1))
--------------------------------------------------------------------------------
V= 1 ka= 0 la= 0 nu= 2 rho= triv
 HH=   (T^10 - T^9 + T^8 - T^5 + T^4 - T^3 + T^2)/((T^12 - 1)*(T - 1))
MMdual= 0
HHc= (T^10 - T^9 + T^8 - T^5 + T^4 - T^3 + T^2)/((T^12 - 1)*(T - 1))
--------------------------------------------------------------------------------
V= 1 ka= 0 la= 0 nu= 0 rho= sgn
 HH=   (T^11 - T^10 + T^9 - T^7 + T^5 - T^4 + T^3)/(T^13 - T^12 - T + 1)
MMdual= 0
HHc= (T^11 - T^10 + T^9 - T^7 + T^5 - T^4 + T^3)/(T^13 - T^12 - T + 1)
--------------------------------------------------------------------------------
V= 1 ka= 0 la= 0 nu= 1 rho= sgn
 HH=   (T^9 - T^8 + T^7 - T^2 + T)/(T^13 - T^12 - T + 1)
MMdual= 0
HHc= (T^9 - T^8 + T^7 - T^2 + T)/(T^13 - T^12 - T + 1)
--------------------------------------------------------------------------------
V= 1 ka= 0 la= 0 nu= 2 rho= sgn
 HH=   (T^7 - T^6 + T^5 + T - 1)/(T^13 - T^12 - T + 1)
MMdual= 1
HHc= (T^13 - T^12 + T^7 - T^6 + T^5)/(T^13 - T^12 - T + 1)
--------------------------------------------------------------------------------
V= 2 ka= 0 la= 0 nu= 0 rho= stan
 HH=   T^4/(T^7 - T^6 - T + 1)
MMdual= 0
HHc= T^4/(T^7 - T^6 - T + 1)
--------------------------------------------------------------------------------
V= 2 ka= 0 la= 0 nu= 1 rho= stan
 HH=   (T^5 - T^3 + T^2)/(T^7 - T^6 - T + 1)
MMdual= 0
HHc= (T^5 - T^3 + T^2)/(T^7 - T^6 - T + 1)
--------------------------------------------------------------------------------
V= 2 ka= 0 la= 0 nu= 2 rho= stan
 HH=   (T^6 - T^5 + T^3)/(T^7 - T^6 - T + 1)
MMdual= 0
HHc= (T^6 - T^5 + T^3)/(T^7 - T^6 - T + 1)
--------------------------------------------------------------------------------
V= 3 ka= 3 la= 3 nu= 0 rho= triv
 HH=   (T^12 + T^8 - T^7 + T^6 + T^2)/(T^13 - T^12 - T + 1)
MMdual= 0
HHc= (T^12 + T^8 - T^7 + T^6 + T^2)/(T^13 - T^12 - T + 1)
--------------------------------------------------------------------------------
V= 3 ka= 1 la= 1 nu= 0 rho= triv
 HH=   (T^12 - T^11 + T^10 + T^6 + T^4)/(T^13 - T^12 - T + 1)
MMdual= 0
HHc= (T^12 - T^11 + T^10 + T^6 + T^4)/(T^13 - T^12 - T + 1)
--------------------------------------------------------------------------------
V= 3 ka= 5 la= 5 nu= 0 rho= triv
 HH=   (T^10 + T^8 + T^4 - T^3 + T^2)/(T^13 - T^12 - T + 1)
MMdual= 0
HHc= (T^10 + T^8 + T^4 - T^3 + T^2)/(T^13 - T^12 - T + 1)
--------------------------------------------------------------------------------
V= 3 ka= 3 la= 3 nu= 0 rho= sgn
 HH=   (T^11 - T^10 + T^9 + T^7 + T^5 - T^4 + T^3)/(T^13 - T^12 - T + 1)
MMdual= 0
HHc= (T^11 - T^10 + T^9 + T^7 + T^5 - T^4 + T^3)/(T^13 - T^12 - T + 1)
```

```
------------------------------------------------------------------------
V= 3 ka= 1 la= 1 nu= 0 rho= sgn
 HH=   (T^11 + T^9 - T^8 + T^7 + T^3 - T^2 + T)/(T^13 - T^12 - T + 1)
MMdual= 0
HHc= (T^11 + T^9 - T^8 + T^7 + T^3 - T^2 + T)/(T^13 - T^12 - T + 1)
------------------------------------------------------------------------
V= 3 ka= 5 la= 5 nu= 0 rho= sgn
 HH=   (T^11 + T^7 - T^6 + T^5 + T^3 + T - 1)/(T^13 - T^12 - T + 1)
MMdual= 1
HHc= (T^13 - T^12 + T^11 + T^7 - T^6 + T^5 + T^3)/(T^13 - T^12 - T + 1)
------------------------------------------------------------------------
V= 6 ka= 0 la= 1 nu= 0 rho= triv
 HH=   T^3/(T^5 - 2*T^4 + T^3 + T^2 - 2*T + 1)
MMdual= 0
HHc= T^3/(T^5 - 2*T^4 + T^3 + T^2 - 2*T + 1)
------------------------------------------------------------------------
V= 6 ka= 0 la= 2 nu= 0 rho= triv
 HH=   T^2/(T^3 - T^2 - T + 1)
MMdual= 0
HHc= T^2/(T^3 - T^2 - T + 1)
------------------------------------------------------------------------
V= 6 ka= 1 la= 2 nu= 0 rho= triv
 HH=   (T^4 - T^2 + T)/(T^5 - 2*T^4 + T^3 + T^2 - 2*T + 1)
MMdual= 0
HHc= (T^4 - T^2 + T)/(T^5 - 2*T^4 + T^3 + T^2 - 2*T + 1)
------------------------------------------------------------------------
V= 6 ka= 3 la= 4 nu= 0 rho= triv
 HH=   (T^4 - T^3 + 2*T - 1)/(T^5 - 2*T^4 + T^3 + T^2 - 2*T + 1)
MMdual= 1
HHc= (T^5 - T^4 + T^2)/(T^5 - 2*T^4 + T^3 + T^2 - 2*T + 1)
------------------------------------------------------------------------
```

In [144…]
```
rat(sum)
```

Out[144]:
```
(T^3 + 8*T^2 + 8*T + 1)/(T^2 - 2*T + 1)
```

In [ ]:
```
Mollie
```

In [146…]
```
Moli=MolienF(6,0,1,0,"triv")
```

In [149…]
```
latex(rat(Moli[0]))
```

Out[149]:
```
\frac{3 T^{18} - 8 T^{17} + 3 T^{16} + 45 T^{15} - 76 T^{14} + 7 T^{13} + 1
34 T^{12} - 90 T^{11} - 103 T^{10} + 195 T^{9} - 44 T^{8} - 76 T^{7} + 58 T
^{6} + 14 T^{5} - 23 T^{4} + 9 T^{3}}{T^{30} - 5 T^{29} + 8 T^{28} + T^{27}
- 19 T^{26} + 22 T^{25} - 3 T^{24} - 7 T^{23} - 5 T^{22} - 4 T^{21} + 49 T^
{20} - 61 T^{19} + 2 T^{18} + 51 T^{17} - 33 T^{16} + 6 T^{15} - 33 T^{14}
+ 51 T^{13} + 2 T^{12} - 61 T^{11} + 49 T^{10} - 4 T^{9} - 5 T^{8} - 7 T^
{7} - 3 T^{6} + 22 T^{5} - 19 T^{4} + T^{3} + 8 T^{2} - 5 T + 1}
```

In [150…]
```
latex(Moli[1])
```

Out[150]:
```
9 T^{3} + 22 T^{4} + 52 T^{5} + 133 T^{6} + 322 T^{7} + 670 T^{8} + 1367 T^
{9} + 2562 T^{10} + 4661 T^{11} + 8131 T^{12} + 13849 T^{13} + 22672 T^{14}
+ 36386 T^{15} + 56700 T^{16} + 86819 T^{17} + 130066 T^{18} + 191986 T^{1
9} + \mathcal{O}\left(T^{20}\right)
```

In [151…]
```
FFF=T^3/(T^5 - 2*T^4 + T^3 + T^2 - 2*T + 1)
```

In [157…]
```
latex(SR(Moli[0]-FFF).series(T))
```

Out[157]:
```
8 T^{3} + 20 T^{4} + 49 T^{5} + 130 T^{6} + 319 T^{7} + 667 T^{8} + 1363 T^
{9} + 2557 T^{10} + 4655 T^{11} + 8125 T^{12} + 13843 T^{13} + 22666 T^{14}
+ 36379 T^{15} + 56692 T^{16} + 86810 T^{17} + 130057 T^{18} + 191977 T^{1
9} + \mathcal{O}\left(T^{20}\right)
```

In [ ]: