```
def Maxwells_Equations_in_Geometric_Calculus():
    Print_Function()
    X = symbols('t x y z')
    (g0,g1,g2,g3,grad) = MV.setup('gamma*t|x|y|z',metric='[1,-1,-1,-1]',coords=X)
    I = MV.I
    B = MV('B','vector',fct=True)
    E = MV('E','vector',fct=True)
    B.set_coef(1,0,0)
    E.set_coef(1,0,0)
    B *= g0
    E *= g0
    J = MV('J','vector',fct=True)
    F = E+I*B
    print r'\text{Pseudo Scalar\;\;}I =',I
    print '\\text{Magnetic Field Bi-Vector\\;\\;} B = \\bm{B\\gamma_{t}} =',B
    print '\\text{Electric Field Bi-Vector\\;\\;} E = \\bm{E\\gamma_{t}} =',E
    print '\\text{Electromagnetic Field Bi-Vector\\;\\;} F = E+IB =',F
    print '%\\text{Four Current Density\\;\\;} J =',J
    gradF = grad*F
    print '#Geometric Derivative of Electomagnetic Field Bi-Vector'
    gradF.Fmt(3,'grad*F')
    print '#Maxwell Equations'
    print 'grad*F = J'
    print '#Div $E$ and Curl $H$ Equations'
    (gradF.grade(1)-J).Fmt(3,'%\\grade{\\nabla F}_{1} -J = 0')
    print '#Curl $E$ and Div $B$ equations'
    (gradF.grade(3)).Fmt(3,'%\\grade{\\nabla F}_{3} = 0')
    return
```

Code Output:

Pseudo Scalar $I = \gamma_t \wedge \gamma_x \wedge \gamma_y \wedge \gamma_z$

Magnetic Field Bi-Vector $B = \boldsymbol{B\gamma_t} = -B^x\gamma_t \wedge \gamma_x - B^y\gamma_t \wedge \gamma_y - B^z\gamma_t \wedge \gamma_z$

Electric Field Bi-Vector $E = \boldsymbol{E\gamma_t} = -E^x\gamma_t \wedge \gamma_x - E^y\gamma_t \wedge \gamma_y - E^z\gamma_t \wedge \gamma_z$

Electromagnetic Field Bi-Vector $F = E + IB = -E^x\gamma_t \wedge \gamma_x - E^y\gamma_t \wedge \gamma_y - E^z\gamma_t \wedge \gamma_z - B^z\gamma_x \wedge \gamma_y + B^y\gamma_x \wedge \gamma_z - B^x\gamma_y \wedge \gamma_z$

Four Current Density $J = J^t\gamma_t + J^x\gamma_x + J^y\gamma_y + J^z\gamma_z$

Geometric Derivative of Electomagnetic Field Bi-Vector Maxwell Equations

$$\boldsymbol{\nabla} F = J$$

Div $E$ and Curl $H$ Equations Curl $E$ and Div $B$ equations

```
def Dirac_Equation_in_Geometric_Calculus():
    Print_Function()
    vars = symbols('t x y z')
    (g0,g1,g2,g3,grad) = MV.setup('gamma*t|x|y|z',metric='[1,-1,-1,-1]',coords=vars)
    I = MV.I
    (m,e) = symbols('m e')
    psi = MV('psi','spinor',fct=True)
    A = MV('A','vector',fct=True)
    sig_z = g3*g0
    print '\\text{4-Vector Potential\\;\\;}\\bm{A} =',A
    print '\\text{8-component real spinor\\;\\;}\\bm{\\psi} =',psi
    dirac_eq = (grad*psi)*I*sig_z-e*A*psi-m*psi*g0
    dirac_eq.simplify()
    dirac_eq.Fmt(3,r'%\text{Dirac Equation\;\;}\nabla \bm{\psi} I \sigma_{z}-e\bm{A}\bm{\psi}-m\bm{\psi}\gamma_{t} = 0')
    return
```

Code Output:

4-Vector Potential $\boldsymbol{A} = A^{t}\gamma_{t} + A^{x}\gamma_{x} + A^{y}\gamma_{y} + A^{z}\gamma_{z}$

8-component real spinor $\boldsymbol{\psi} = \psi + \psi^{tx}\gamma_{t}\wedge\gamma_{x} + \psi^{ty}\gamma_{t}\wedge\gamma_{y} + \psi^{tz}\gamma_{t}\wedge\gamma_{z} + \psi^{xy}\gamma_{x}\wedge\gamma_{y} + \psi^{xz}\gamma_{x}\wedge\gamma_{z} + \psi^{yz}\gamma_{y}\wedge\gamma_{z} + \psi^{txyz}\gamma_{t}\wedge\gamma_{x}\wedge\gamma_{y}\wedge\gamma_{z}$

```
def Lorentz_Tranformation_in_Geometric_Algebra():
    Print_Function()
    (alpha,beta,gamma) = symbols('alpha beta gamma')
    (x,t,xp,tp) = symbols("x t x' t'")
    (g0,g1) = MV.setup('gamma*t|x',metric='[1,-1]')
    from sympy import sinh,cosh
    R = cosh(alpha/2)+sinh(alpha/2)*(g0^g1)
    X = t*g0+x*g1
    Xp = tp*g0+xp*g1
    print 'R =',R
    print r"#%t\bm{\gamma_{t}}+x\bm{\gamma_{x}} = t'\bm{\gamma'_{t}}+x'\bm{\gamma'_{x}} = R\lp t'\bm{\gamma_{t}}+x'\bm{\gamma_{x}}\rp R^{\dagger}"
    Xpp = R*Xp*R.rev()
    Xpp = Xpp.collect()
    Xpp = Xpp.subs({2*sinh(alpha/2)*cosh(alpha/2):sinh(alpha),sinh(alpha/2)**2+cosh(alpha/2)**2:cosh(alpha)})
    print r"%t\bm{\gamma_{t}}+x\bm{\gamma_{x}} =",Xpp
    Xpp = Xpp.subs({sinh(alpha):gamma*beta,cosh(alpha):gamma})
    print r'%\f{\sinh}{\alpha} = \gamma\beta'
    print r'%\f{\cosh}{\alpha} = \gamma'
    print r"%t\bm{\gamma_{t}}+x\bm{\gamma_{x}} =",Xpp.collect()
    return
```

Code Output:

$$R = \cosh\left(\frac{\alpha}{2}\right) + \sinh\left(\frac{\alpha}{2}\right)\gamma_{t}\wedge\gamma_{x}$$

$$t\boldsymbol{\gamma_{t}} + x\boldsymbol{\gamma_{x}} = t'\boldsymbol{\gamma'_{t}} + x'\boldsymbol{\gamma'_{x}} = R\left(t'\boldsymbol{\gamma_{t}} + x'\boldsymbol{\gamma_{x}}\right)R^{\dagger}$$

$$t\boldsymbol{\gamma_{t}} + x\boldsymbol{\gamma_{x}} = \left(2t'\sinh^{2}\left(\frac{\alpha}{2}\right) + t' - x'\sinh(\alpha)\right)\gamma_{t} + \left(-t'\sinh(\alpha) + 2x'\sinh^{2}\left(\frac{\alpha}{2}\right) + x'\right)\gamma_{x}$$

$$\sinh(\alpha) = \gamma\beta$$

$$\cosh(\alpha) = \gamma$$

$$t\boldsymbol{\gamma_{t}} + x\boldsymbol{\gamma_{x}} = \left(-\beta\gamma x' + 2t'\sinh^{2}\left(\frac{\alpha}{2}\right) + t'\right)\gamma_{t} + \left(-\beta\gamma t' + 2x'\sinh^{2}\left(\frac{\alpha}{2}\right) + x'\right)\gamma_{x}$$