LAB - 9 : Computation of basis, dimension for a vector space and graphical representation of linear transformation.

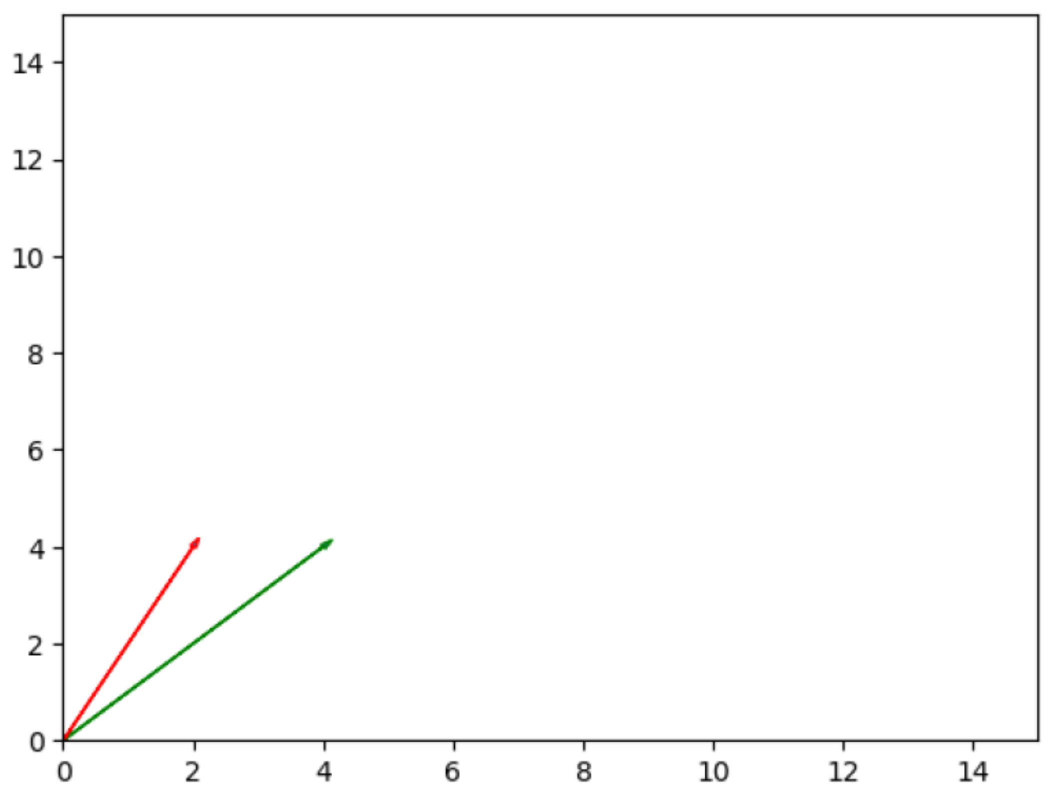
```
# Verification of Rank and nullity theorem
from numpy import *
A= Matrix([[1,2,3],[4,5,6],[7,8,9]])
r=A.rank()
print('Rank of the linear transformation : r = ',r)
NullSpace = A.nullspace()
print('Null space of the linear transformation :\n ')
NullSpace = Matrix(NullSpace)
pprint(NullSpace)
n=NullSpace.shape[1]
#print('Nullity of the linear transformation : n = ',n)
dim=int(input('Enter the dimension of the vector space U :'))
if dim==r+n:
    print('Rank and Nullity theorem holds good \n dim(u)= dim(R(T))+dim(N(T
else:
    print('dim(u)!= dim(R(T))+dim(N(T))')
```

OUTPUT :

```
Rank of the linear transformation : r =  2
Null space of the linear transformation :

⎡1 ⎤
⎢  ⎥
⎢-2⎥
⎢  ⎥
⎣1 ⎦
Enter the dimension of the vector space U :3
Rank and Nullity theorem holds good
 dim(u)= dim(R(T))+dim(N(T))
```

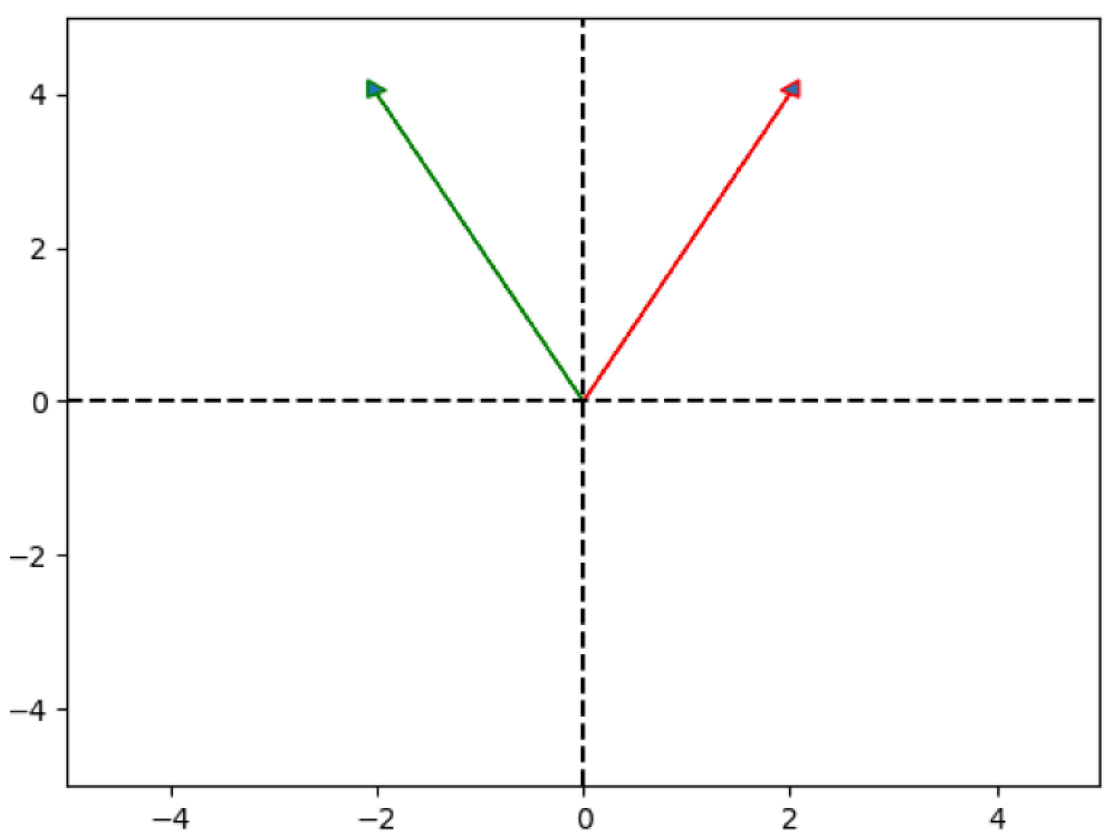
```python
# HORIZONTAL STRETCH T:R^2 to R^2 such that T(x,y)=(2x,y)
from matplotlib.pyplot import *
x=2
y=4
X=2*x
Y=y
# Creating our arrow
arrow(0,0,X,Y,head_width=0.1, head_length=0.2,ec='g')
arrow(0,0, x,y, head_width=0.1, head_length=0.2,ec='r')
# X and Y coordinates
ylim(0,15)
xlim(0,15)
show()
```

OUTPUT :



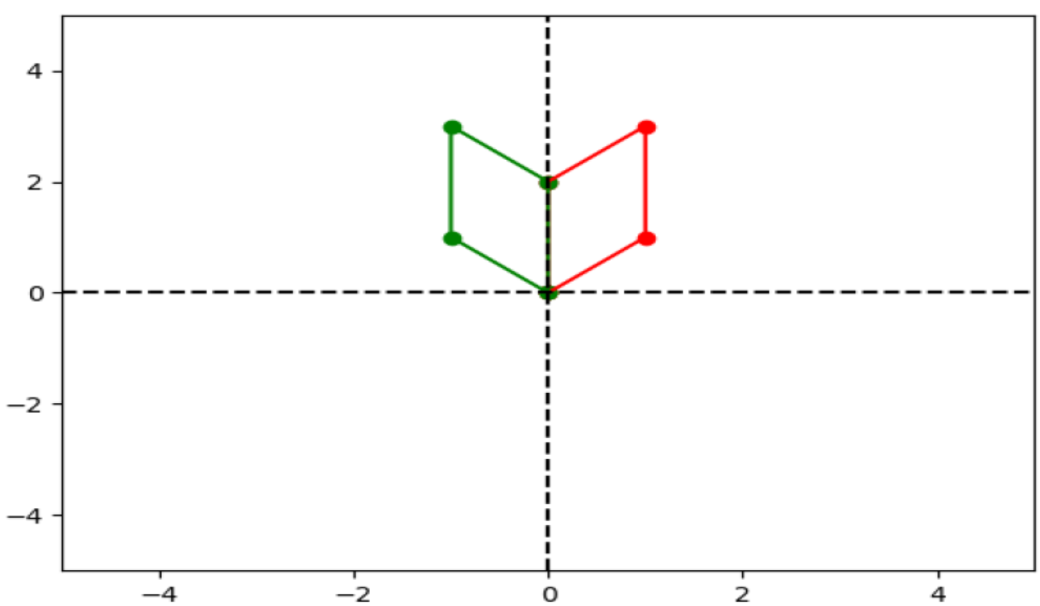
```
# REFLECTION T:R^2 to R^2 such that T(x,y)=(-x,y)
from matplotlib.pyplot import *
x=2
y=4
X=-1*x
Y=1*y
# Creating our arrow
arrow(0,0,X,Y,head_width=0.2, head_length=0.2,ec='g')
arrow(0,0, x,y, head_width=0.2, head_length=0.2,ec='r')
# X and Y coordinates
ylim(-5,5)
xlim(-5,5)
axvline(x=0,color='k',ls='--')
axhline(y=0,color='k',ls='--')
show()
```

OUTPUT :



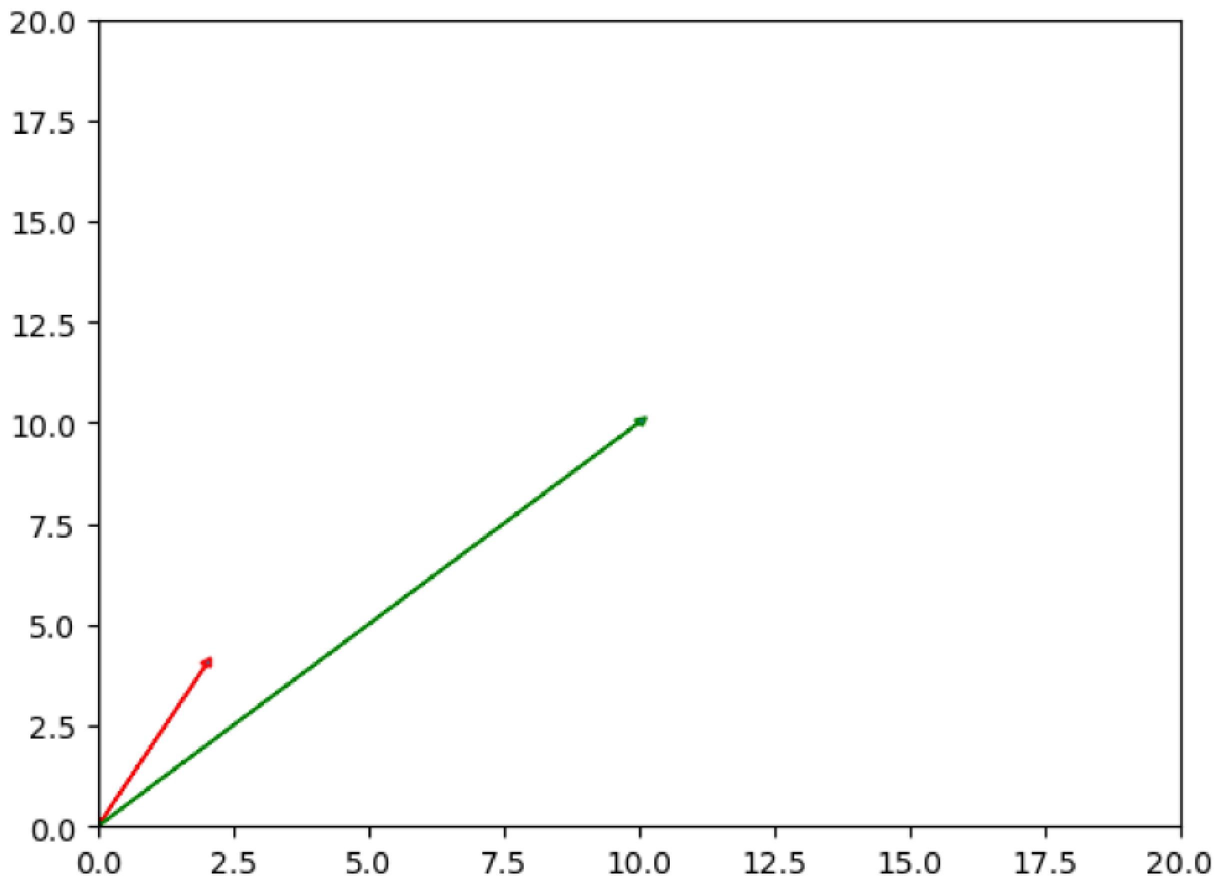
```python
#REFLECTION T:R^2 to R^2 such that T(x,y)=(-x,y)
from numpy import *
from matplotlib.pyplot import *
x=[0,1,1,0,0]
y=[0,1,3,2,0]
X=zeros(len(x))
Y=zeros(len(x))
plot(x,y,'r-')
for i in range(len(x)):
    X[i]=-1*x[i]
    Y[i]=y[i]
scatter(x,y,color='r')
scatter(X,Y,color='g')
plot(X,Y,'g-')
# X and Y coordinates
ylim(-5,5)
xlim(-5,5)
axvline(x=0,color='k',ls='--')
axhline(y=0,color='k',ls='--')
show()
```

OUTPUT :



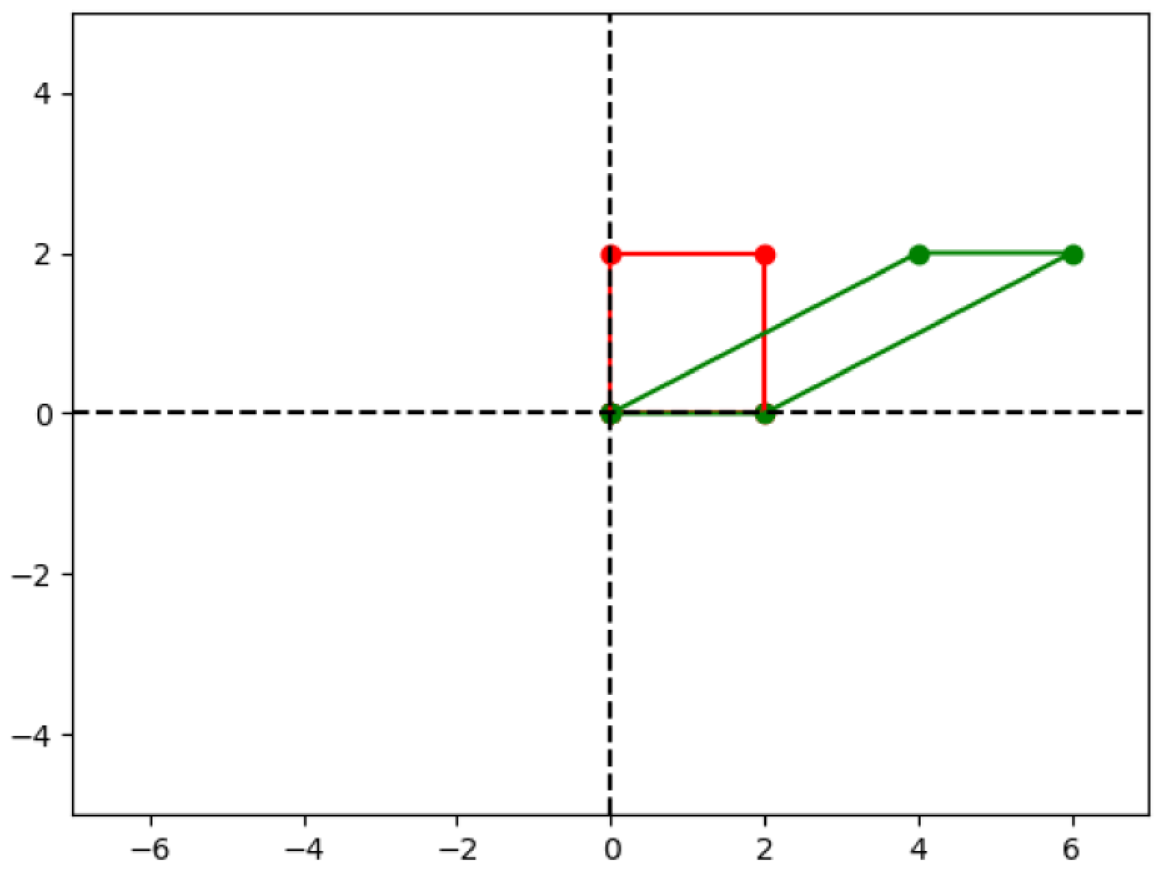
```python
# SHEAR TRANSFORMATION T:R^2 to R^2 such that T(x,y)=(x+2y,y+3x)
from matplotlib.pyplot import *
x=2
y=4
X=x+2*y
Y=y+3*x
# Creating our arrow
arrow(0,0, x,y, head_width=0.2, head_length=0.2,ec='r')
arrow(0,0,X,Y,head_width=0.2, head_length=0.2,ec='g')
# X and Y coordinates
ylim(0,20)
xlim(0,20)
show()
```

OUTPUT :



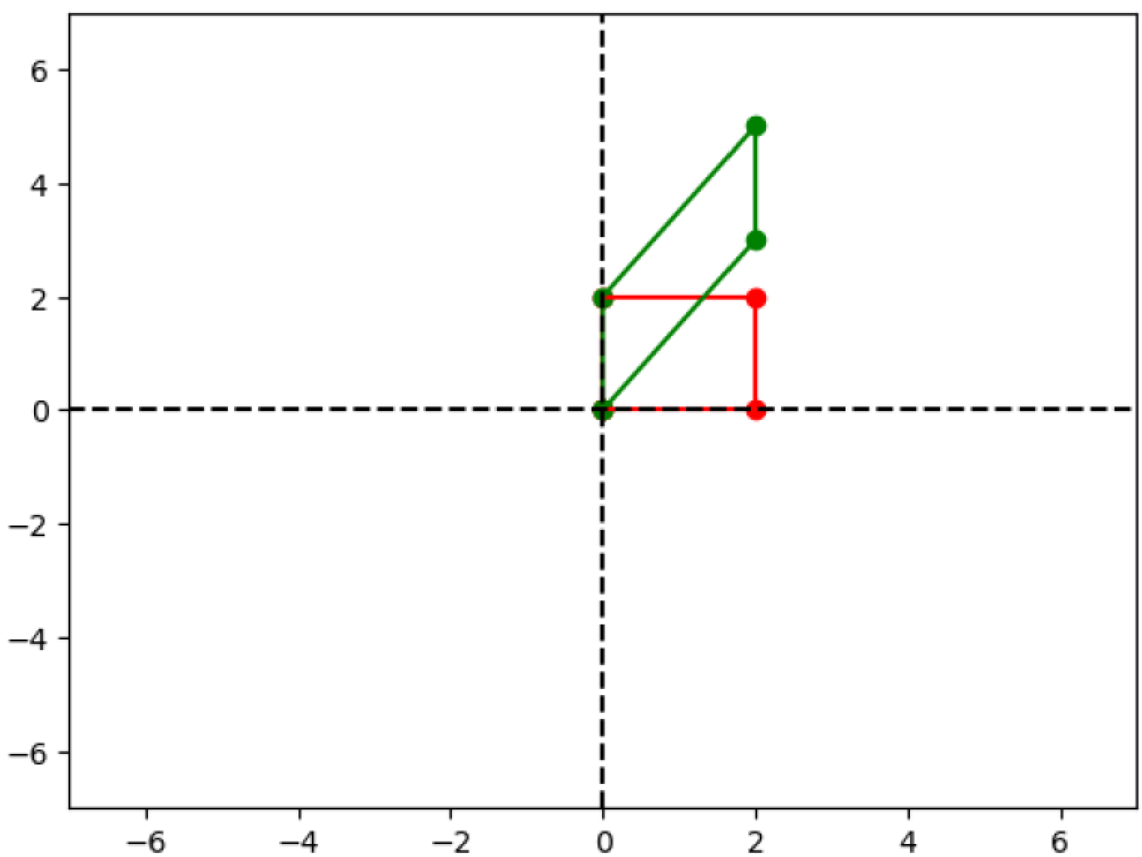
```python
# HORIZONTAL SHEAR TRANSFORMATION T:R^2 to R^2 such that T(x,y)=(x+2y,y)
from matplotlib.pyplot import *
x=[0,2,2,0,0]
y=[0,0,2,2,0]
X=zeros(len(x))
Y=zeros(len(x))
plot(x,y,'r-')
for i in range(len(x)):
    X[i]=x[i]+(2)*y[i]
    Y[i]=y[i]
scatter(x,y,color='r')
scatter(X,Y,color='g')
plot(X,Y,'g-')
# X and Y coordinates
ylim(-5,5)
xlim(-7,7)
axvline(x=0,color='k',ls='--')
axhline(y=0,color='k',ls='--')
show()
```

OUTPUT :



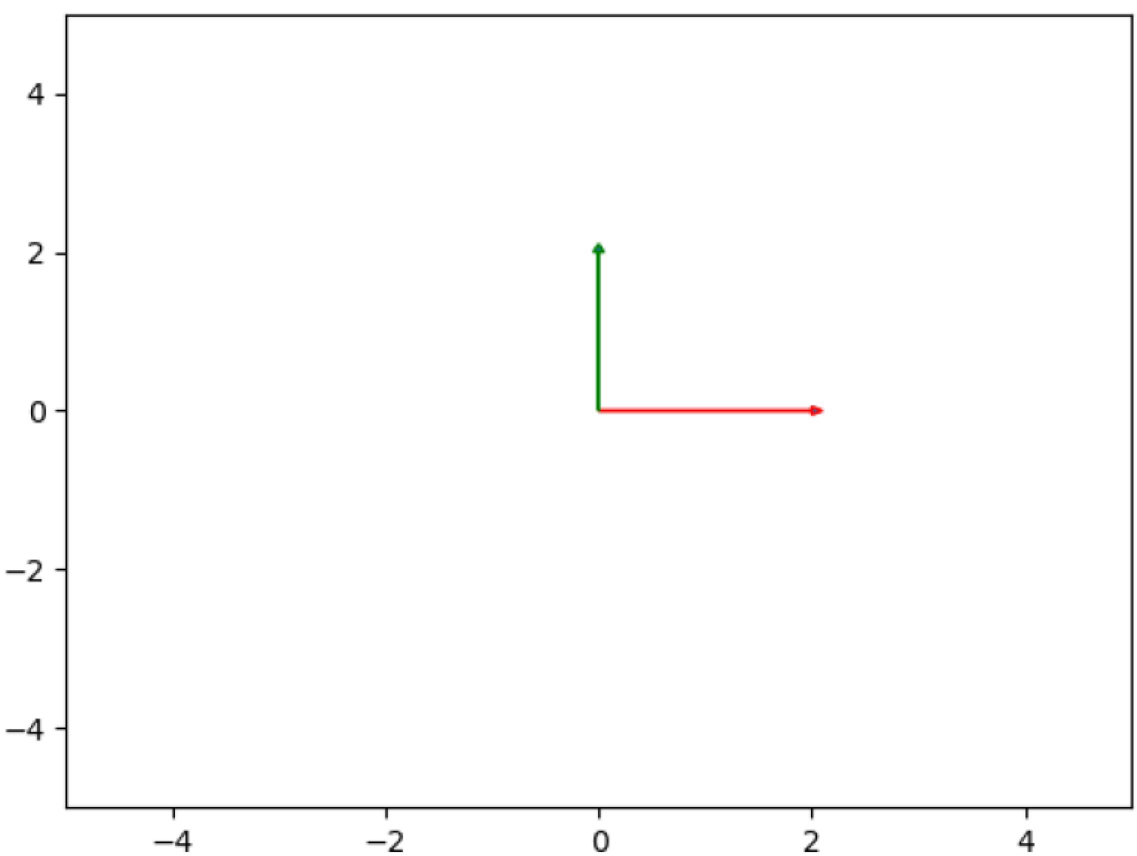
```
# VERTICAL SHEAR TRANSFORMATION T:R^2 to R^2 such that T(x,y)=(x,y+1.5x)
from matplotlib.pyplot import *
x=[0,2,2,0,0]
y=[0,0,2,2,0]
X=zeros(len(x))
Y=zeros(len(x))
plot(x,y,'r-')
for i in range(len(x)):
    X[i]=x[i]
    Y[i]=y[i]+1.5*x[i]
scatter(x,y,color='r')
scatter(X,Y,color='g')
plot(X,Y,'g-')
# X and Y coordinates
ylim(-7,7)
xlim(-7,7)
axvline(x=0,color='k',ls='--')
axhline(y=0,color='k',ls='--')
show()
```

OUTPUT :



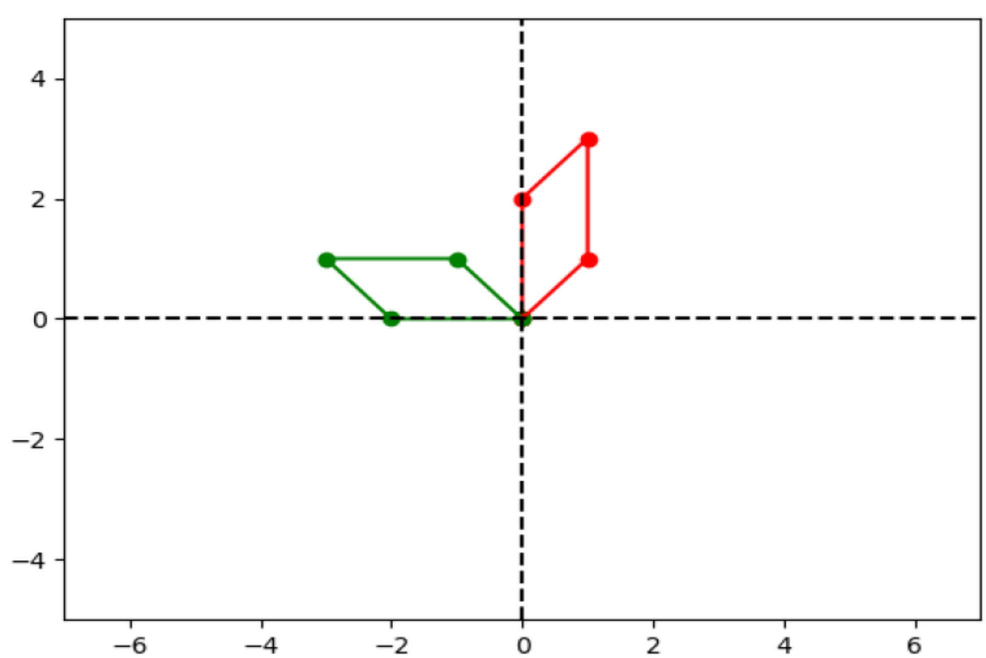
```
# ROTATION T:R^2 to R^2 such that T(x,y)=(-y,x)
from matplotlib.pyplot import *
x=2
y=0
X=-y
Y=x
# Creating our arrow
arrow(0,0, x,y, head_width=0.1, head_length=0.1,ec='r')
arrow(0,0,X,Y,head_width=0.1, head_length=0.1,ec='g')

# X and Y coordinates
ylim(-5,5)
xlim(-5,5)
show()
```

OUTPUT :



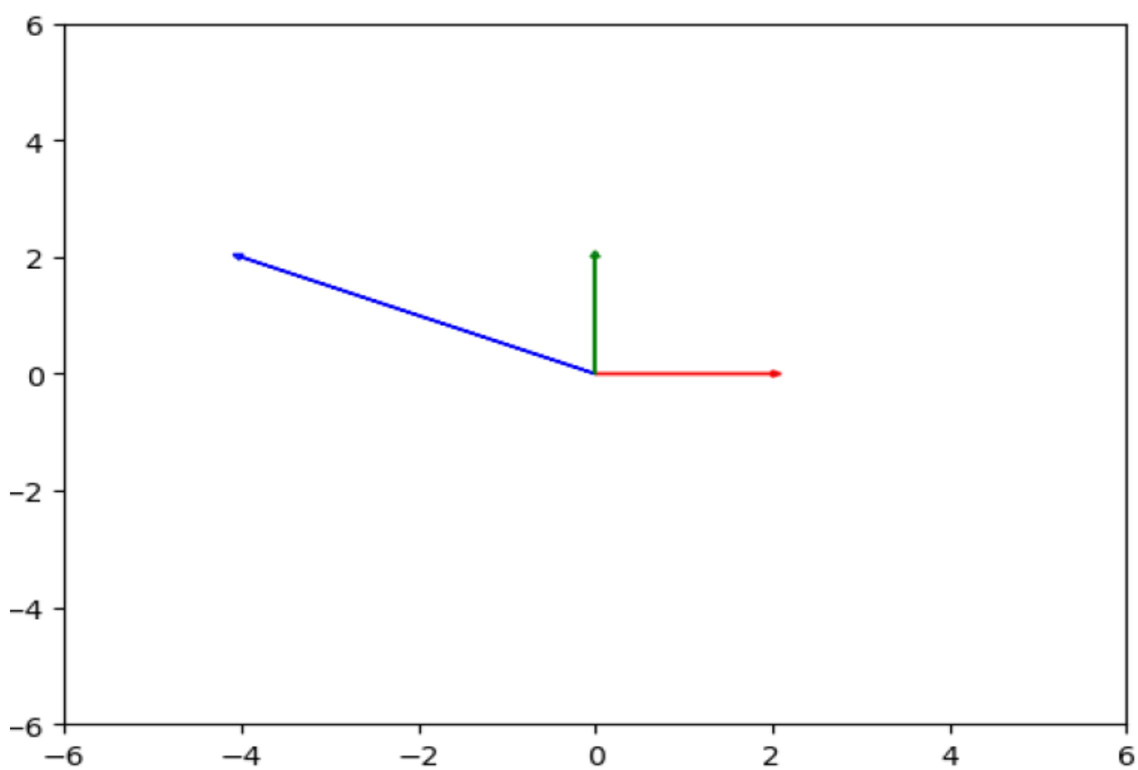
```
# ROTATION T:R^2 to R^2 such that T(x,y)=(-y,x)
from matplotlib.pyplot import *
x=[0,1,1,0,0]
y=[0,1,3,2,0]
X=zeros(len(x))
Y=zeros(len(x))
plot(x,y,'r-')
for i in range(len(x)):
    X[i]=-1*y[i]
    Y[i]=x[i]
scatter(x,y,color='r')
scatter(X,Y,color='g')
plot(X,Y,'g-')
# X and Y coordinates
ylim(-5,5)
xlim(-7,7)
axvline(x=0,color='k',ls='--')
axhline(y=0,color='k',ls='--')
show()
```

OUTPUT :



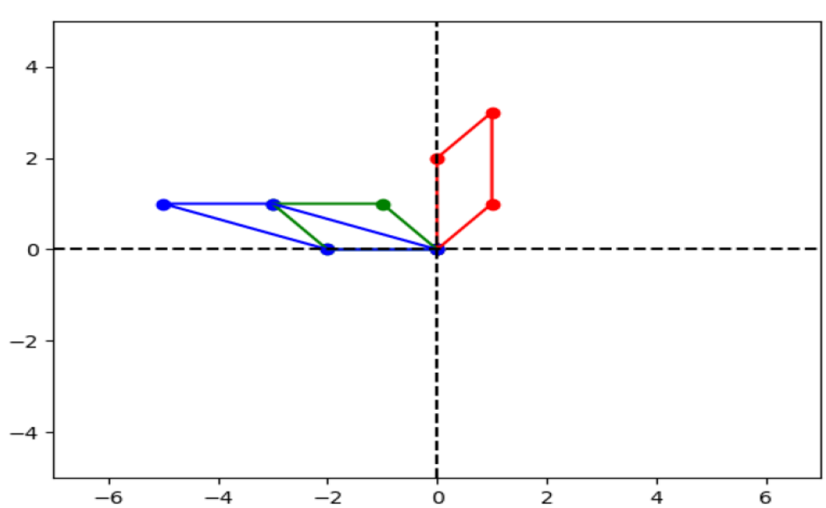
```python
# COMPOSITION TRANSFORMATION (ROTATION AND HORIZONTAL SHEAR)
from matplotlib.pyplot import *
x=2
y=0
X=-y
Y=x
X1=X-2*Y
Y1=Y
# Creating our arrow
arrow(0,0, x,y, head_width=0.1, head_length=0.1,ec='r')
arrow(0,0,X1,Y1,head_width=0.1, head_length=0.1,ec='b')
arrow(0,0,X,Y,head_width=0.1, head_length=0.1,ec='g')
# X and Y coordinates
ylim(-6,6)
xlim(-6,6)
show()
```

OUTPUT :



```python
# COMPOSITION TRANSFORMATION (ROTATION AND HORIZONTAL SHEAR)
from matplotlib.pyplot import *
x=[0,1,1,0,0]
y=[0,1,3,2,0]
X=zeros(len(x))
Y=zeros(len(x))
X1=zeros(len(x))
Y1=zeros(len(x))
plot(x,y,'r-')
for i in range(len(x)):
    X[i]=-1*y[i]
    Y[i]=x[i]
    X1[i]=X[i]-2*Y[i]
    Y1[i]=Y[i]
scatter(x,y,color='r')
scatter(X,Y,color='g')
scatter(X1,Y1,color='b')
plot(X,Y,'g-')
plot(X1,Y1,'b-')
# X and Y coordinates
ylim(-5,5)
xlim(-7,7)
axvline(x=0,color='k',ls='--')
axhline(y=0,color='k',ls='--')
show()
```

OUTPUT :



```python
# Finding the Basis for the subspace of a vector space R3
from sympy import *
from numpy import *
A=array([[0,0,4],[0,1,1],[2,3,4],[0,2,4]])
m=A.shape[0]
n=A.shape[1]
for i in range(n):
    if A[i,i]==0:
        for j in range(i+1,m):
            if A[j,i]!=0:
                A[[i,j]]=A[[j,i]]
    for i1 in range(i+1,m):
        k=A[i1,i]/A[i,i]
        for j1 in range(i,n):
            A[i1,j1]=A[i1,j1]-k*A[i,j1]
    B=Matrix(A)
    pprint(B)
    print('\n')
BS=A[~all(A==0,axis=1)]
pprint(Matrix(BS))
print('\n Basis for the subspace is {',end=' ')
for i in range(len(BS)):
    print(BS[i],end=' ')
print('}')
print('Dimension of the subspace is ',len(BS))
```

$$\begin{bmatrix} 2 & 3 & 4 \\ 0 & 1 & 1 \\ 0 & 0 & 4 \\ 0 & 2 & 4 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 3 & 4 \\ 0 & 1 & 1 \\ 0 & 0 & 4 \\ 0 & 0 & 2 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 3 & 4 \\ 0 & 1 & 1 \\ 0 & 0 & 4 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 3 & 4 \\ 0 & 1 & 1 \\ 0 & 0 & 4 \end{bmatrix}$$

```
 Basis for the subspace is { [2 3 4] [0 1 1] [0 0 4] }
Dimension of the subspace is  3
```

Exercise

1. Using python program to verify the rank-nullity theorem for the linear transformation

   $T: R^3 \to R^3$ defined by $T(x, y, z) = (x + 4y + 7z, 2x + 5y + 8z, 3x + 6y + 9z)$

2. Using python program to verify the rank-nullity theorem for the linear transformation

   $T: R^3 \to R^3$ defined by $T(x, y, z) = (x + y, x - y, 2x - z)$

3. Using python program to verify the rank-nullity theorem for the linear transformation

   $T: R^3 \to R^3$ defined by $T(x, y, z) = (x + y, y + z, z + x)$

4. Using python program to verify the rank-nullity theorem for the linear transformation

   $T: R^3 \to R^3$ defined by $T(x, y, z) = (x + y + z, 2x + 3z, x + 2y + 4z)$

5. Using python program find the image of the vector $(5,0)$ when it is rotated by 90^0 then stretched horizontally.

6.
   a) Using python program to find the image of vector $(2,3)$ when it is stretched horizontally.
   b) Using python program to find the image of vector $(4,0)$ when it is rotated by 90^0.

7.
   a) Using python program to find the image of vector $(2,4)$ when it is stretched vertically.
   b) Using python program to find the image of vector $(3,3)$ when it is reflected about y-axis.

8. Using python program to verify the rank-nullity theorem for the linear transformation

$T: R^3 \rightarrow R^3$ defined by $T(x, y, z) = (x - y + 2z, y, x + 2y + z)$

9. Using python program to
   a) Find the image of vector $(3,4)$ when it is reflected about y-axis.

   b) Find the image of vector $(0,5)$ when it is rotated by 90^0.

10. Using python program to

    a) Find the image of vector $(3,3)$ when it is stretched horizontally.

    b) Find the image of vector $(4,5)$ when it is reflected about y-axis.