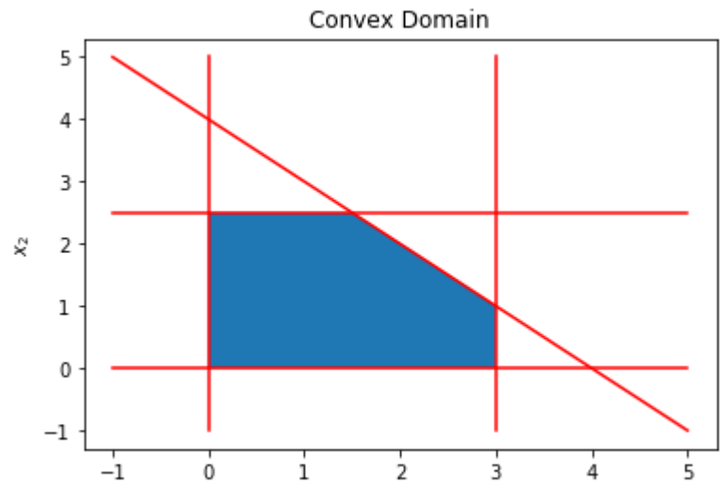
In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

# Visualisation

In [2]:
```
theta=np.linspace(-1,5,10)
f_theta=4*np.ones_like(theta)-theta
plt.plot(theta,f_theta,'-r')
plt.plot(theta,2.5*np.ones_like(theta),'-r')
plt.plot(theta,np.zeros_like(theta),'-r')
plt.plot(np.zeros_like(theta),theta,'-r')
plt.plot(3*np.ones_like(theta),theta,'-r')
plt.fill([0,3,3,1.5,0],[0,0,1,2.5,2.5])
plt.title('Convex Domain')
plt.xlabel('$x_1$')
plt.ylabel('$x_2$')
plt.show()
```




# Closed Form Projection

In [3]:
```
def projectionPi(p):
    p[0]=max(min(p[0],3),0)
    p[1]=max(min(p[1],2.5),0)


    a=np.array([1.5,2.5])
    b=np.array([3,1])
    if 1.5<=p[0]<= 3 and 1<=p[1]<=2.5 and p[0]+p[1]>4:
        p=a+(np.dot(np.transpose(p-a),(b-a))/np.linalg.norm(b-a)**2)*(b-a)

    return p
```

# Gradient Descent

In [4]:
```
def gradientDescent(x,lr):
    grad=np.array([2*(x[0]-2), 2*(2*x[1]-7)])
    x=projectionPi(x-lr*grad)

    return x
```

In [5]:
```
x=np.array([2.5,1])
for i in range(2):
    x=gradientDescent(x,0.05)
    print(x)
```

```
[2.45 1.5 ]
[2.2525 1.7475]
```