24Fall Advanced Control for Robotics

Homework 1

Name: Siyuan Wang SID: 12443028

Problem 1 - Python Basics

- a.

In [1]:

```
from datetime import datetime

now = datetime.now()

print("current date and time: ", now)
```

```
current date and time:  2024-09-26 01:45:28.155770
```

- b.

In [2]:

```
simple_list = ['aa', 'bb', 'cc', 'dd', 'ee', 'ff', 'gg']
remove_index = [0, 4, 5]

for i in reversed(remove_index):
    simple_list.pop(i)

simple_list
```

Out[2]:

```
['bb', 'cc', 'dd', 'gg']
```

- c.

In [3]:

```
class Student():
    def __init__(self, name, age):
        self.name = name
        self.age = age
    def print_info(self):
        print("Student Name: {}".format(self.name))
        print("Student Age: {}".format(self.age))
```

```
weizhang = Student(name='weizhang', age=18)
weizhang.print_info()
```

```
Student Name: weizhang
Student Age: 18
```

Problem 2 - Linear Algebra

- a.

In [4]:

```
import numpy as np

A = np.array([
    [1, -2, 4],
    [1, -1, 1],
    [1, 0, 0],
    [1, 1, 1]
])
B = np.array([
    [1, 2, 3],
    [1, 2, 3],
    [1, 2, 3],
    [1, 2, 3],
])

A, B
```

Out[4]:

```
(array([[ 1, -2,  4],
        [ 1, -1,  1],
        [ 1,  0,  0],
        [ 1,  1,  1]]),
 array([[1, 2, 3],
        [1, 2, 3],
        [1, 2, 3],
        [1, 2, 3]]))
```

- b.

In [5]:

```
print("The second row of A:", A[1, :])
print("The second row of B:", B[:, 2])
```

```
The second row of A: [ 1 -1  1]
The second row of B: [3 3 3 3]
```

- c.

In [6]:

```
print("A + B = \n", A+B)
print("A - B = \n", A-B)
```

```
A + B =
 [[2 0 7]
 [2 1 4]
 [2 2 3]
 [2 3 4]]
A - B =
 [[ 0 -4  1]
 [ 0 -3 -2]
 [ 0 -2 -3]
 [ 0 -1 -2]]
```

- d.

In [7]:

```
res = np.concatenate( (A, B), axis=1)
res
```

Out[7]:

```
array([[ 1, -2,  4,  1,  2,  3],
       [ 1, -1,  1,  1,  2,  3],
       [ 1,  0,  0,  1,  2,  3],
       [ 1,  1,  1,  1,  2,  3]])
```

- e.

In [8]:

```
A.transpose() @ B
```

Out[8]:

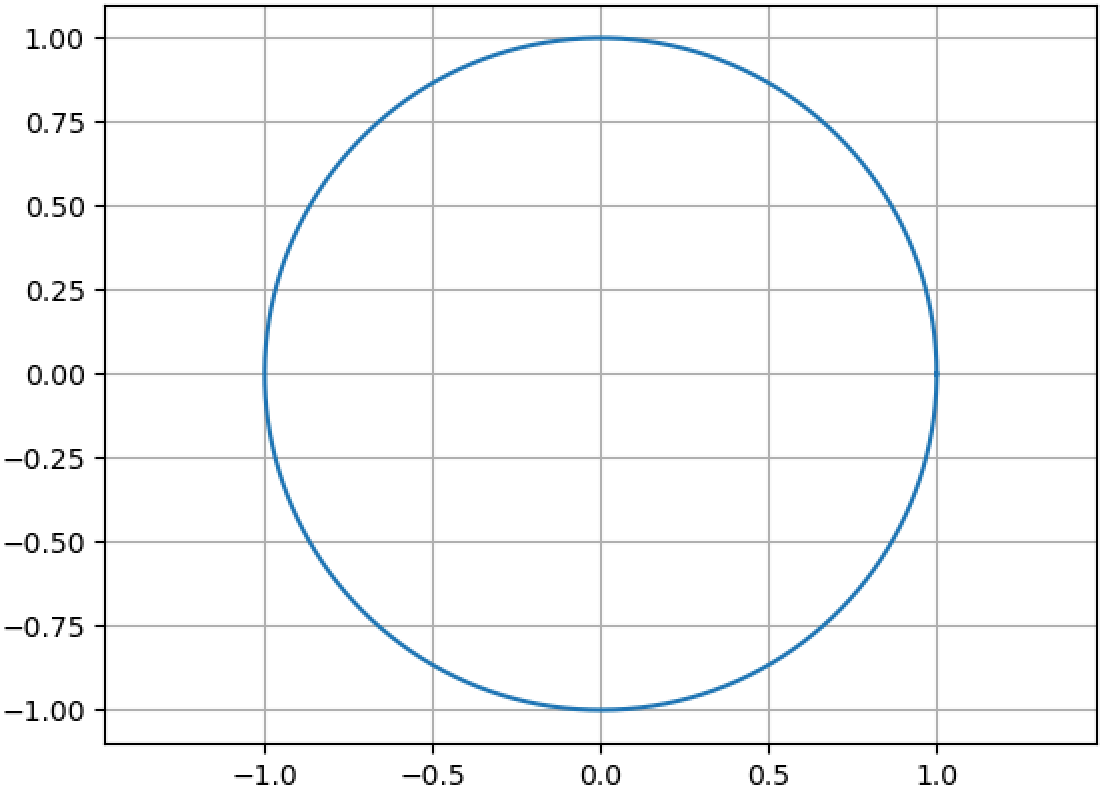
```
array([[ 4,  8, 12],
       [-2, -4, -6],
       [ 6, 12, 18]])
```

Problem 3 - Matplotlib

- a.

In [9]:

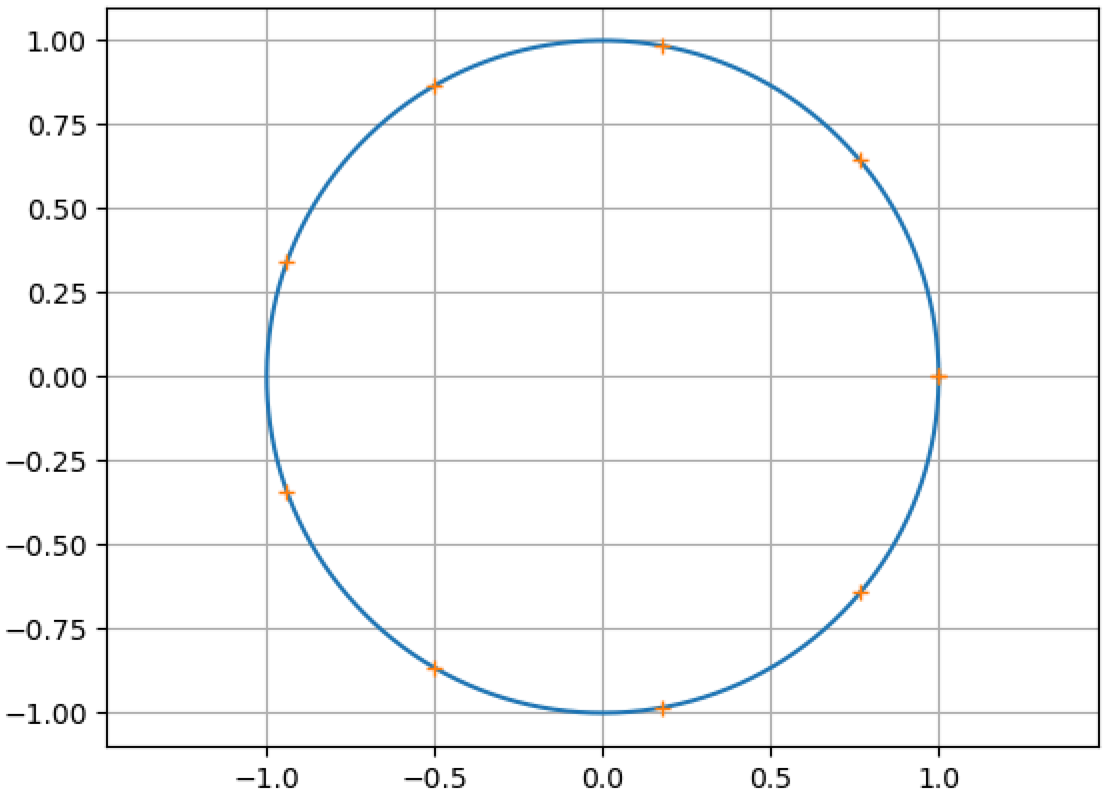
```
import matplotlib.pyplot as plt

def plot_unit_circle():
    p = np.linspace(0, 2*np.pi, 1000)
    x = np.cos(p)
    y = np.sin(p)

    plt.plot(x, y)
    plt.axis('equal')
    plt.grid()

plot_unit_circle()
```



- b.

```
In [10]: plot_unit_circle()

         plus_signs = np.linspace(0, 2*np.pi, 10)
         x_plus_signs = np.cos(plus_signs)
         y_plus_signs = np.sin(plus_signs)
         plt.plot(x_plus_signs, y_plus_signs, '+')
```

```
Out[10]: [<matplotlib.lines.Line2D at 0x7faa2f6677c0>]
```



Problem 7 - Ellipsoids

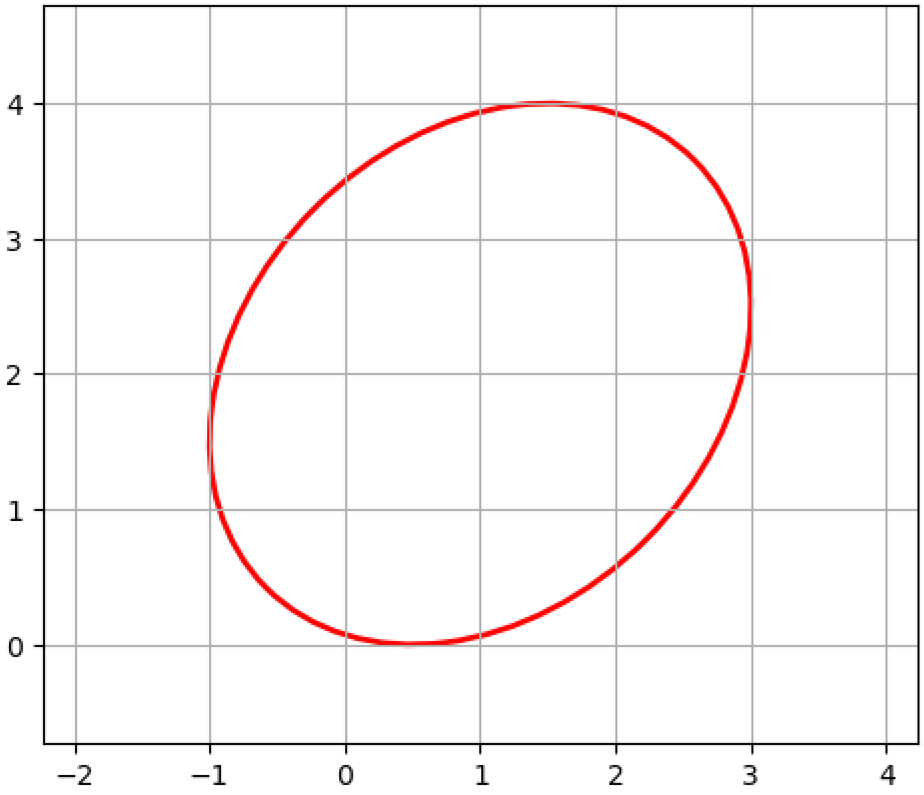
- 3.

In [11]:

```python
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.patches import Ellipse

def draw_ellipse(center, major_axis, minor_axis, a, b, fig_padding=1):
    # 中心点
    center_x, center_y = center

    # 长轴和短轴的方向向量
    major_axis_dx, major_axis_dy = major_axis
    minor_axis_dx, minor_axis_dy = minor_axis

    # 计算旋转角度
    angle = np.degrees(np.arctan2(major_axis_dy, major_axis_dx))

    # 创建椭圆
    ellipse = Ellipse(xy=(center_x, center_y), width=2*a, height=2*b, angle=angle,

    # 创建图形
    fig, ax = plt.subplots()
    ax.add_patch(ellipse)

    # 设置图形的范围
    ax.set_xlim(center_x - a - fig_padding, center_x + a + fig_padding)
    ax.set_ylim(center_y - b - fig_padding, center_y + b + fig_padding)
    plt.grid()

    # 显示图形
    plt.gca().set_aspect('equal', adjustable='box')
    plt.show()
```

In [12]:

```
# Plotting Ellipsoid
center = (1, 2)  # 椭圆中心
major_axis = (1, 1)  # 长轴方向向量
minor_axis = (1, -1)  # 短轴方向向量
a = np.sqrt(5)  # 长半轴长度
b = np.sqrt(3)  # 短半轴长度

draw_ellipse(center, major_axis, minor_axis, a, b)
```



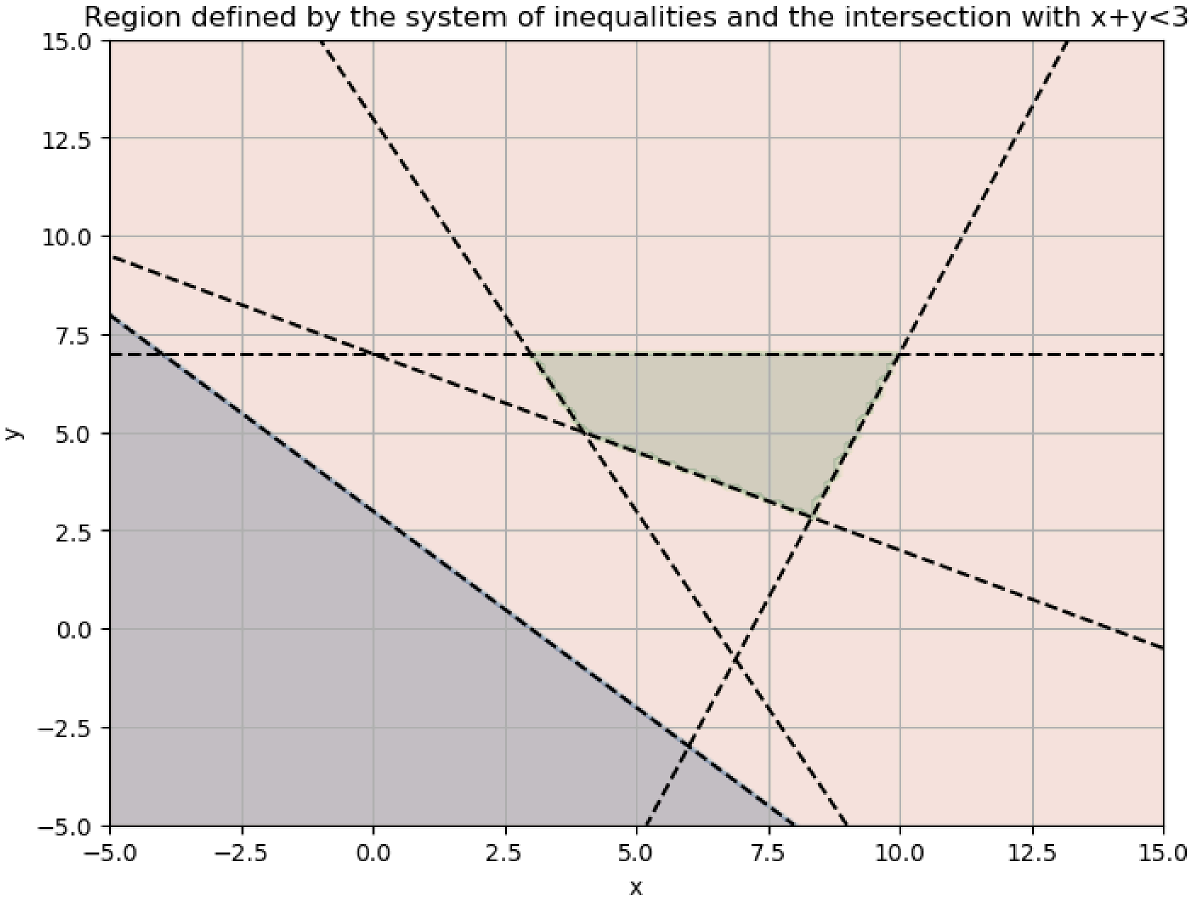
Problem 8 - Polyhedron

- 2.

In [13]:

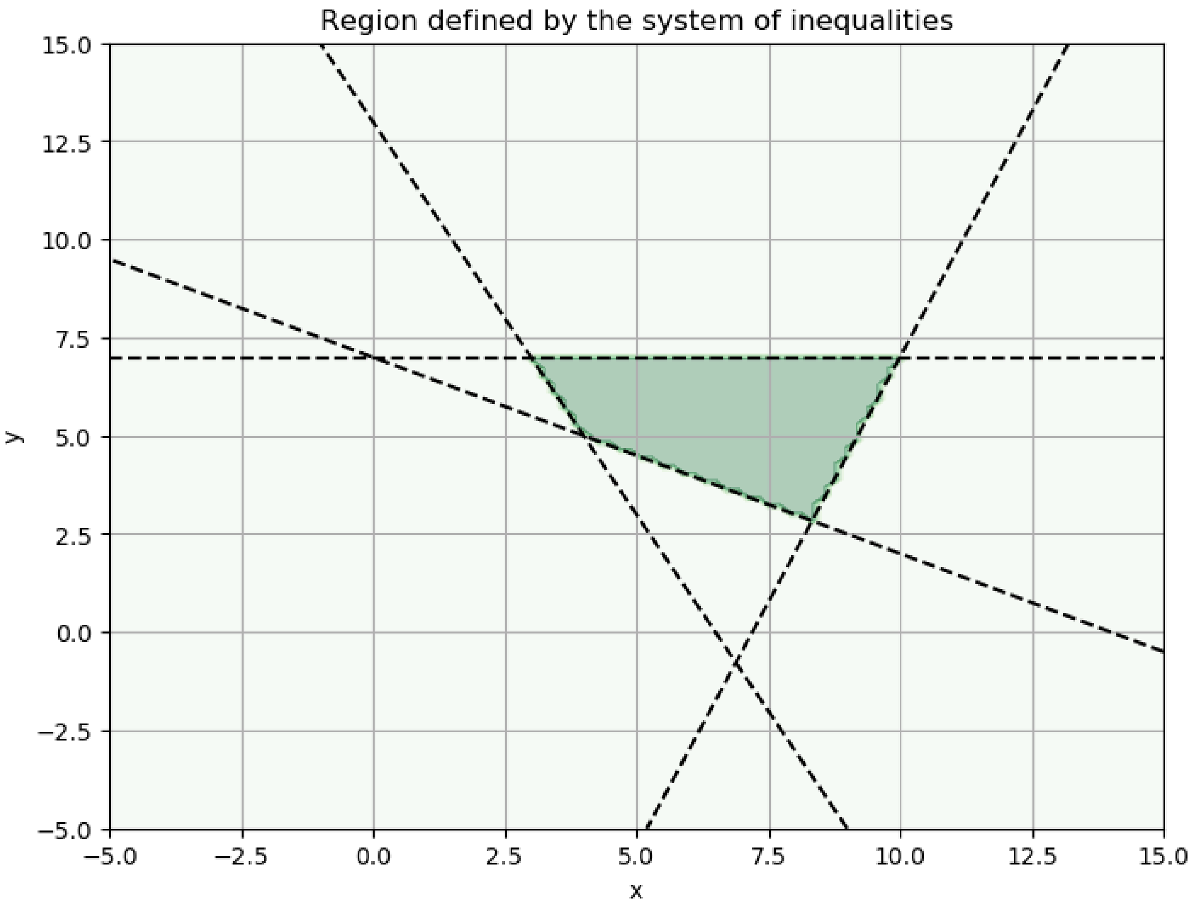
```
import numpy as np
import matplotlib.pyplot as plt

# 原始线性不等式的系数矩阵 A 和向量 b
A = np.array([
    [0, 1],  # 第一个不等式 2x - 3y < 5
    [5, -2], # 第二个不等式 x + y < 4
    [-1, -2],# 第三个不等式 -x + 2y < 3
    [-4, -2] # 第四个不等式 3x - y < 7
])
b = np.array([7, 36, -14, -26])
```

```
# 新增不等式的系数和右侧常数
A_new = np.array([[1, 1]])  # 不等式 x + y < 3
b_new = np.array([3])

# 创建坐标网格
x_min, x_max = -5, 15
y_min, y_max = -5, 15
xx, yy = np.meshgrid(np.linspace(x_min, x_max, 100), np.linspace(y_min, y_max, 100)

# 初始化布尔数组来存储满足所有原始不等式的点
Z = np.ones_like(xx, dtype=bool)

# 对于每一个原始不等式，计算哪些点满足它
for i in range(A.shape[0]):
    Z &= (A[i, 0] * xx + A[i, 1] * yy < b[i])

# 计算满足新不等式的点
Z_new = (A_new[0, 0] * xx + A_new[0, 1] * yy < b_new[0])

# 计算两个区域的交集
Z_intersection = Z & Z_new

# 绘制图形
plt.figure(figsize=(8, 6))
# 填充原始不等式的区域
plt.contourf(xx, yy, Z, alpha=0.3, cmap='Greens')
# 填充新不等式的区域
plt.contourf(xx, yy, Z_new, alpha=0.3, cmap='Blues')
# 填充两者的交集区域，使用不同的颜色
plt.contourf(xx, yy, Z_intersection, alpha=0.3, cmap='Reds')

# 绘制每个不等式的边界线
for i in range(A.shape[0]):
    plt.contour(xx, yy, A[i, 0] * xx + A[i, 1] * yy, [b[i]], colors='k', linestyles
# 绘制新增不等式的边界线
plt.contour(xx, yy, A_new[0, 0] * xx + A_new[0, 1] * yy, [b_new[0]], colors='k', li

# 设置坐标轴范围
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)

# 添加坐标轴标签
plt.xlabel('x')
plt.ylabel('y')

# 显示图形
plt.title('Region defined by the system of inequalities and the intersection with x
plt.grid(True)
plt.show()
```



In [14]:

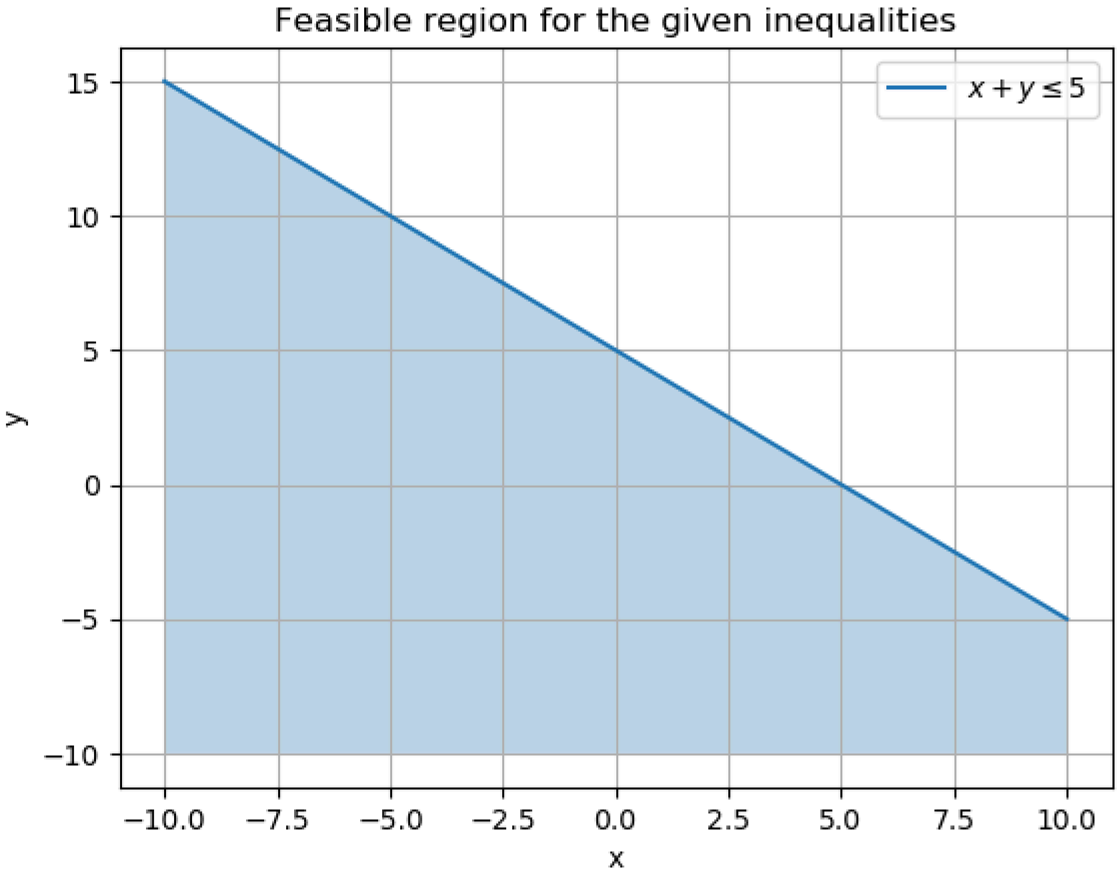
```
import numpy as np
import matplotlib.pyplot as plt

A = np.array([
    [ 0, 1 ],
    [ 5, -2],
    [-1, -2],
    [-4, -2]
])
b = np.array([7, 36, -14, -26])

x_min, x_max = -5, 15
y_min, y_max = -5, 15
xx, yy = np.meshgrid(np.linspace(x_min, x_max, 100), np.linspace(y_min, y_max, 100)

Z = np.ones_like(xx, dtype=bool)

for i in range(A.shape[0]):
    Z &= (A[i, 0] * xx + A[i, 1] * yy < b[i])

plt.figure(figsize=(8, 6))
plt.contourf(xx, yy, Z, alpha=0.3, cmap='Greens')  # 填充满足所有不等式的区域
for i in range(A.shape[0]):
    plt.contour(xx, yy, A[i, 0] * xx + A[i, 1] * yy, [b[i]], colors='k', linestyles
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)

plt.xlabel('x')
plt.ylabel('y')

plt.title('Region defined by the system of inequalities')
plt.grid(True)

plt.show()
```



In [15]:

```
import numpy as np
import matplotlib.pyplot as plt

a = np.array([1, 1])
b = 5

x = np.linspace(-10, 10, 400)

y = (b - a[0] * x) / a[1]

fig, ax = plt.subplots()

ax.plot(x, y, label=r'$x + y \leq 5$')

ax.fill_between(x, y, -10, where=y >= -10, alpha=0.3)

ax.legend()
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.grid()
ax.set_title('Feasible region for the given inequalities')
```

```
Out[15]:  Text(0.5, 1.0, 'Feasible region for the given inequalities')
```


Feasible region for the given inequalities
$x + y \leq 5$
y
x