1.5.34

RAVULA SHASHANK REDDY - EE25BTECH11047

August 29, 2025

Question

The point P which divides the line segment joining the points A (2, -5) and B (5,2) in the ratio 2 : 3 lies in which quadrant?

Equation

The formula for internal division of vectors is where P divides A and B in the ratio k:1

$$\mathbf{P} = \frac{k\mathbf{B} + \mathbf{A}}{1 + k}$$

Theoretical Solution

Let O be the origin. Then the position vectors

$$\mathbf{OA} = \begin{pmatrix} 2 \\ -5 \end{pmatrix}, \quad \mathbf{OB} = \begin{pmatrix} 5 \\ 2 \end{pmatrix}.$$

The point P, dividing the segment AB in the ratio $2:3$ internally, has the position vector

$$\mathbf{OP} = \frac{2\mathbf{OB} + 3\mathbf{OA}}{3+2} = \frac{2\begin{pmatrix} 5 \\ 2 \end{pmatrix} + 3\begin{pmatrix} 2 \\ -5 \end{pmatrix}}{5} = \frac{\begin{pmatrix} 10 \\ 4 \end{pmatrix} + \begin{pmatrix} 6 \\ -15 \end{pmatrix}}{5} = \frac{\begin{pmatrix} 16 \\ -11 \end{pmatrix}}{5}.$$

Therefore the co-ordinates of P are

$$\left(\frac{16}{5}, -\frac{11}{5}\right).$$

Since $x > 0$ and $y < 0$, P lies in the **IV (fourth) quadrant.**

C Code - Section formula function

```c
// section_formula.c
#include <stdio.h>

void find_section_point(double x1, double y1, double x2, double
    y2, double m, double n, double* x, double* y) {
        *x = (m * x2 + n * x1) / (m + n);
        *y = (m * y2 + n * y1) / (m + n);
}
```

Python Code through shared output

```
# Section Formula Problem

import numpy as np
import matplotlib.pyplot as plt

# Given points
A = np.array(([2, -5])).reshape(-1,1)
B = np.array(([5, 2])).reshape(-1,1)

# Ratio m:n = 2:3
m, n = 2, 3

# Point dividing AB in ratio m:n
P = (n*A + m*B) / (m+n)

# Determine Quadrant
x, y = P[0,0], P[1,0]
```

```
if x > 0 and y > 0:
    quadrant = First Quadrant
elif x < 0 and y > 0:
    quadrant = Second Quadrant
elif x < 0 and y < 0:
    quadrant = Third Quadrant
elif x > 0 and y < 0:
    quadrant = Fourth Quadrant
else:
    quadrant = On Axis

print(fCoordinates of P: ({x:.2f}, {y:.2f}))
print(fP lies in the {quadrant})

# Generate line AB
x_AB = np.linspace(A[0,0], B[0,0], 100)
y_AB = np.linspace(A[1,0], B[1,0], 100)
```

```python
# Plot line AB
plt.plot(x_AB, y_AB, label='$AB$')

# Plot points A, B, P
plt.scatter([A[0,0], B[0,0], P[0,0]], [A[1,0], B[1,0], P[1,0]],
    color='red')
labels = ['A(2,-5)', 'B(5,2)', f'P({x:.2f},{y:.2f})']
for i, txt in enumerate(labels):
    plt.annotate(txt, ( [A[0,0], B[0,0], P[0,0]][i],
                        [A[1,0], B[1,0], P[1,0]][i]),
                textcoords=offset points, xytext=(10,-10))
```

```
# Styling axes
ax = plt.gca()
ax.spines['left'].set_position('zero')
ax.spines['bottom'].set_position('zero')
ax.spines['top'].set_color('none')
ax.spines['right'].set_color('none')

plt.legend(loc='best')
plt.grid(True)
plt.axis('equal')
plt.show()
```

Python code : Direct

```
import numpy as np
import matplotlib.pyplot as plt
#local imports
from libs.line.funcs import *
from libs.triangle.funcs import *
from libs.conics.funcs import circ_gen


# Given points
A = np.array(([2,-5])).reshape(-1,1)
B = np.array(([5,2])).reshape(-1,1)

# Ratio m:n = 2:3
m, n = 2, 3

# Point dividing AB in ratio m:n
P = (n*A + m*B) / (m+n)
```

```python
# Generating line AB
def line_gen(A,B):

    len = 100
    dim = A.shape[0]
    x_AB = np.zeros((dim,len))
    lam_1 = np.linspace(0,1,len)


    for i in range(len):
        temp1 = A + lam_1[i]*(B-A)
        x_AB[:,i]= temp1.T
    return x_AB

x_AB = line_gen(A,B)

# Plotting line AB
plt.plot(x_AB[0,:], x_AB[1,:], label='$AB$')

# Plotting points A, B, P
```

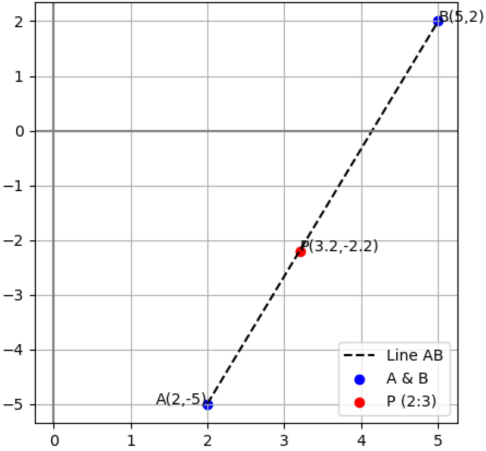
```python
tri_coords = np.block([[A,B,P]])
plt.scatter(tri_coords[0,:], tri_coords[1,:])
vert_labels = ['A','B','P']

for i, txt in enumerate(vert_labels):
    plt.annotate(f'{txt}\n({tri_coords[0,i]:.1f}, {tri_coords[1,i]:.1f})',
                 (tri_coords[0,i], tri_coords[1,i]),
                 textcoords=offset points,
                 xytext=(20,-10), ha='center')

# Axis styling
ax = plt.gca()
ax.spines['left'].set_position('zero')
ax.spines['bottom'].set_position('zero')
ax.spines['top'].set_color('none')
ax.spines['right'].set_color('none')
```

```python
plt.legend(loc='best')
plt.grid()
plt.axis('equal')
plt.show()
```

Plot by python using shared output from c



Plot by python only