1.5.7

EE25BTECH11019 – Darji Vivek M.

Question

If $\left(\frac{a}{3}, 4\right)$ is the midpoint of the line segment joining the points $(-6, 5)$ and $(-2, 3)$, then the value of a is $(10, 2021)$

Variables Used

Symbol	Meaning
A	Point $(-6, 5)$
B	Point $(-2, 3)$
M	Midpoint
a	Unknown to find

Midpoint Formula

$$\mathbf{A} = \begin{pmatrix} -6 \\ 5 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} -2 \\ 3 \end{pmatrix} \tag{1}$$

$$\mathbf{M} = \frac{\mathbf{A} + \mathbf{B}}{2} \tag{2}$$

Substitution

$$\mathbf{M} = \frac{1}{2}\left(\begin{pmatrix}-6\\5\end{pmatrix} + \begin{pmatrix}-2\\3\end{pmatrix}\right) \tag{3}$$

$$= \frac{1}{2}\begin{pmatrix}-8\\8\end{pmatrix} \tag{4}$$

$$= \begin{pmatrix}-4\\4\end{pmatrix} \tag{5}$$

Given Midpoint

$$\mathbf{M} = \begin{pmatrix} \frac{a}{3} \\ 4 \end{pmatrix} \tag{6}$$

Equating components:

$$\frac{a}{3} = -4 \implies a = -12 \tag{7}$$

Final Answer

$$\boxed{a = -12}$$

C Code

```c
#include <stdio.h>

double find_a(int x1, int y1, int x2, int y2, int given_y)
{
    double mid_x = (x1 + x2) / 2.0;
    // given midpoint is (a/4, given_y)
    return 4 * mid_x;
}
```

Python Code (Import and Setup)

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

# Load the shared object file (compiled from your C code)
lib = ctypes.CDLL('./lib1.so')  # change name if your .so file

# Define argument and return types for the C function
lib.find_a.argtypes = [ctypes.c_int, ctypes.c_int,
                       ctypes.c_int, ctypes.c_int,
                       ctypes.c_int]
lib.find_a.restype = ctypes.c_double
```

Python Code (Calling C Function)

```
# Given data
x1, y1 = -6, 5
x2, y2 = -2, 3
given_y = 4

# Call the C function
a_value = lib.find_a(x1, y1, x2, y2, given_y)
print(f"Value of a: {a_value}")

# Midpoint coordinates from a
mid_x = a_value / 4
mid_y = given_y
```

Python Code (Preparing Data)

```
# Create numpy arrays for plotting
A = np.array([x1, y1])
B = np.array([x2, y2])
M = np.array([mid_x, mid_y])

# Generate line between A and B
line_AB = np.column_stack((A, B))
```

Python Code (Plotting)

```
# Plotting
plt.plot([A[0], B[0]], [A[1], B[1]], label='$AB$')
plt.scatter([A[0], B[0], M[0]],
            [A[1], B[1], M[1]],
            color=['red', 'blue', 'green'])

# Annotate points
labels = ['A', 'B', 'Midpoint']
coords = [A, B, M]
for label, coord in zip(labels, coords):
    plt.annotate(f'{label}\n({coord[0]:.2f}, {coord[1]:.2f})',
                 (coord[0], coord[1]),
                 textcoords="offset points",
                 xytext=(10, -10),
                 ha='center')
```

Python Code (Finalizing The Plot)

```
plt.legend()
plt.grid(True)
plt.axis('equal')
plt.savefig('1.png')
plt.show()
```

Plot

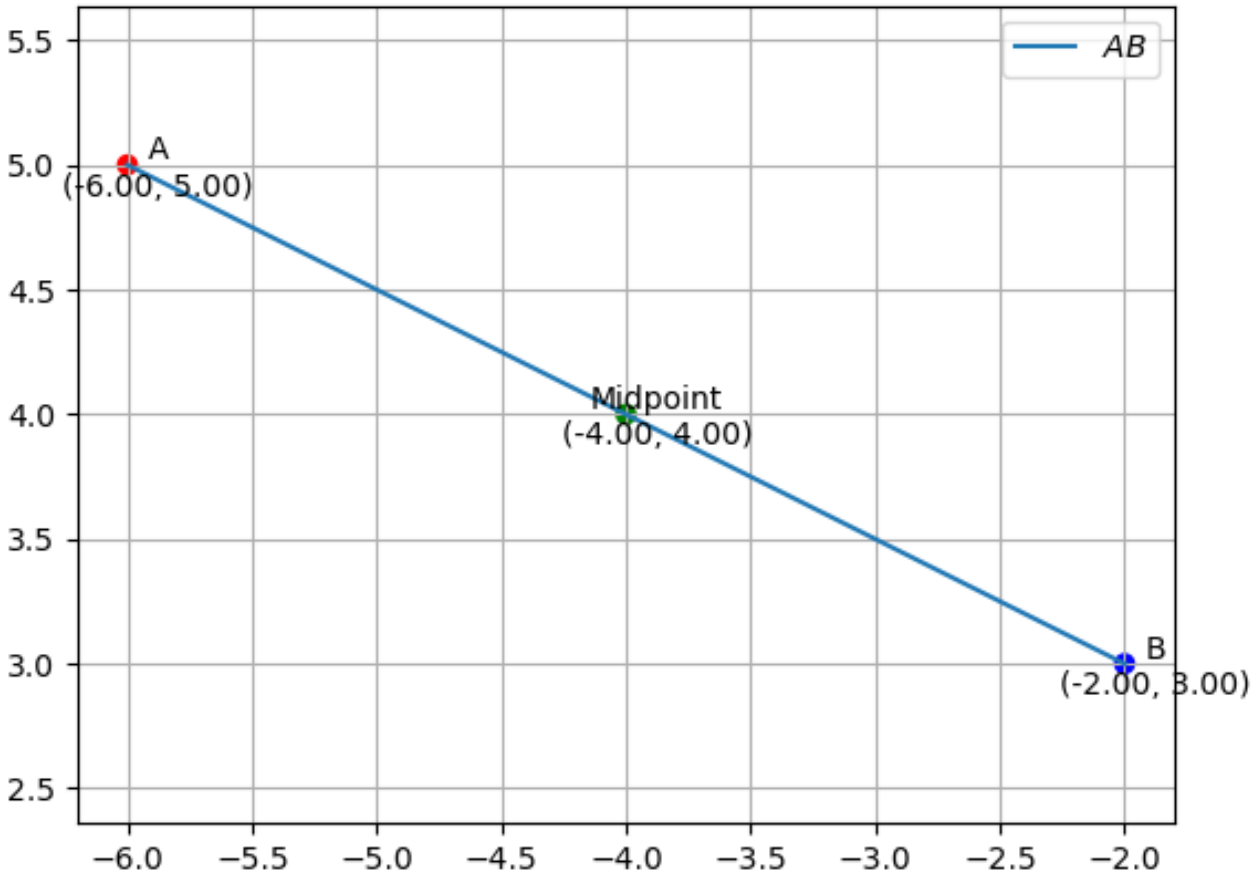

Figure: plot