

4.3.50

Namaswi-EE25BTECH11060

september 14,2025

Question

Find the equation of the lines which makes intercepts -3 and 2 on the x and y axes respectively

Given that line passes through points $(-3, 0)$ and $(0, 2)$

Let

Vector	<i>coordinate</i>
A	$(-3, 0)$
B	$(0, 2)$
n	(a, b)

As equation of line is given by

$$\mathbf{n}^T \mathbf{x} = 1 \quad (1)$$

Solution

So, for A

$$\begin{pmatrix} a \\ b \end{pmatrix}^T \begin{pmatrix} -3 \\ 0 \end{pmatrix} = 1 \quad (2)$$

for B

$$\begin{pmatrix} a \\ b \end{pmatrix}^T \begin{pmatrix} 0 \\ 2 \end{pmatrix} = 1 \quad (3)$$

(4)

Solution

From 2 and 3

$$\begin{pmatrix} -3 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad (5)$$

In augmented matrix form

$$\left[\begin{array}{cc|c} -3 & 0 & 1 \\ 0 & 2 & 1 \end{array} \right] \quad (6)$$

Divide Row 1 by -3

$$\left[\begin{array}{cc|c} -1 & 0 & \frac{1}{3} \\ 0 & 2 & 1 \end{array} \right] \quad (7)$$

Solution

Divide Row 2 by 2

$$\left[\begin{array}{cc|c} -1 & 0 & \frac{-1}{3} \\ 0 & 1 & \frac{1}{2} \end{array} \right] \quad (8)$$

$$a = \frac{-1}{3} \text{ and } b = \frac{1}{2} \quad (9)$$

(10)

So from 1 Equation of line is $\left(\frac{-1}{3} \right) x = 1$

```
#include <stdio.h>
int main() {
    // Given: line passes through A(-3, 0) and B(0, 2)
    float A[2] = {-3.0, 0.0};
    float B[2] = {0.0, 2.0};
    // We want to find normal vector n = (a, b) such that:
    //  $n^T \cdot A = 1 \Rightarrow a \cdot (-3) + b \cdot 0 = 1 \Rightarrow -3a = 1 \Rightarrow a = -1/3$ 
    //  $n^T \cdot B = 1 \Rightarrow a \cdot 0 + b \cdot 2 = 1 \Rightarrow 2b = 1 \Rightarrow b = 1/2$ 
    float a = -1.0 / 3.0;
    float b = 1.0 / 2.0;
```

```
printf("Given Points:\n");
printf("A = (%.1f, %.1f)\n", A[0], A[1]);
printf("B = (%.1f, %.1f)\n\n", B[0], B[1]);
printf("Normal Vector n = (a, b) = (%.2f, %.2f)\n", a, b);
// Equation of line in dot product form:  $a*x + b*y = 1$ 
printf("\nEquation of line in dot product form:\n");
printf("%.2fx + %.2fy = 1\n", a, b);
// Convert to standard form: Multiply both sides by LCM(3,2)
    = 6
// Original:  $-x/3 + y/2 = 1$ 
// Multiply by 6:  $-2x + 3y = 6$  --> Standard form:  $2x - 3y + 6$ 
    = 0
```



```
int A_coeff = 2;
int B_coeff = -3;
int C_coeff = 6;
printf("\nEquation of line in standard form:\n");
printf("%dx ", A_coeff);
if (B_coeff < 0)
    printf("- %dy ", -B_coeff);
else
    printf("+ %dy ", B_coeff);
if (C_coeff < 0)
    printf("- %d = 0\n", -C_coeff);
else
    printf("+ %d = 0\n", C_coeff);
return 0;
}
```

Python Code

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

# Line equation:  $-2x + 3y = 6$ 
# Solve for y:  $y = (2x + 6)/3$ 

# Define x values for plotting
x = np.linspace(-10, 10, 400)
y = (2 * x + 6) / 3

# Find intercepts
# X-intercept: set  $y = 0$   $-2x = 6$   $x = -3$   $A = (-3, 0)$ 
# Y-intercept: set  $x = 0$   $3y = 6$   $y = 2$   $B = (0, 2)$ 
A = (-3, 0)
B = (0, 2)
```

Python Code

```
# Plot the line
plt.plot(x, y, label='Line:  $-2x + 3y = 6$ ', color='blue')

# Mark the intercepts
plt.scatter(*A, color='red', zorder=5)
plt.scatter(*B, color='green', zorder=5)

# Annotate the points
plt.text(A[0]-1, A[1]-0.5, f'A {A}', color='red', fontsize=12)
plt.text(B[0]+0.2, B[1]+0.2, f'B {B}', color='green', fontsize
        =12)

# Axes lines
plt.axhline(0, color='black', linewidth=1)
plt.axvline(0, color='black', linewidth=1)
```

```
# Graph settings
plt.title('Graph of the Line  $-2x + 3y = 6$ ')
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.grid(True)
plt.legend()
plt.axis('equal')
plt.xlim(-10, 10)
plt.ylim(-10, 10)

# Show the plot
plt.show()
```

C and Python Code

```
1 import ctypes
2 import os
3
4 # Load the shared object file
5 lib_path = os.path.abspath("liblineeq.so")
6 lib = ctypes.CDLL(lib_path)
7
8 # Define the function's argument types
9 lib.line_from_intercepts.argtypes = [ctypes.c_double, ctypes.c_double]
10
11 # Optional: Define the return type (void function, so None)
12 lib.line_from_intercepts.restype = None
```

C and Python Code

```
# Example intercepts
x_intercept = -3.0
y_intercept = 2.0

print("Calling C function from Python with:")
print(f" X-intercept = {x_intercept}")
print(f" Y-intercept = {y_intercept}\n")

# Call the function
lib.line_from_intercepts(x_intercept, y_intercept)
```

