Matgeo Presentation - Problem 1.2.10

ai25btech11004 - jaswanth

August 28, 2025

Question

Find the vector joining the points **P**(2,3,0) and **Q**(-1,-2,-4) directed from **P** to **Q**.

Solution

Name	Point
P	$\begin{pmatrix} 2 \\ 3 \\ 0 \end{pmatrix}$
Q	$\begin{pmatrix} -1 \\ -2 \\ -4 \end{pmatrix}$

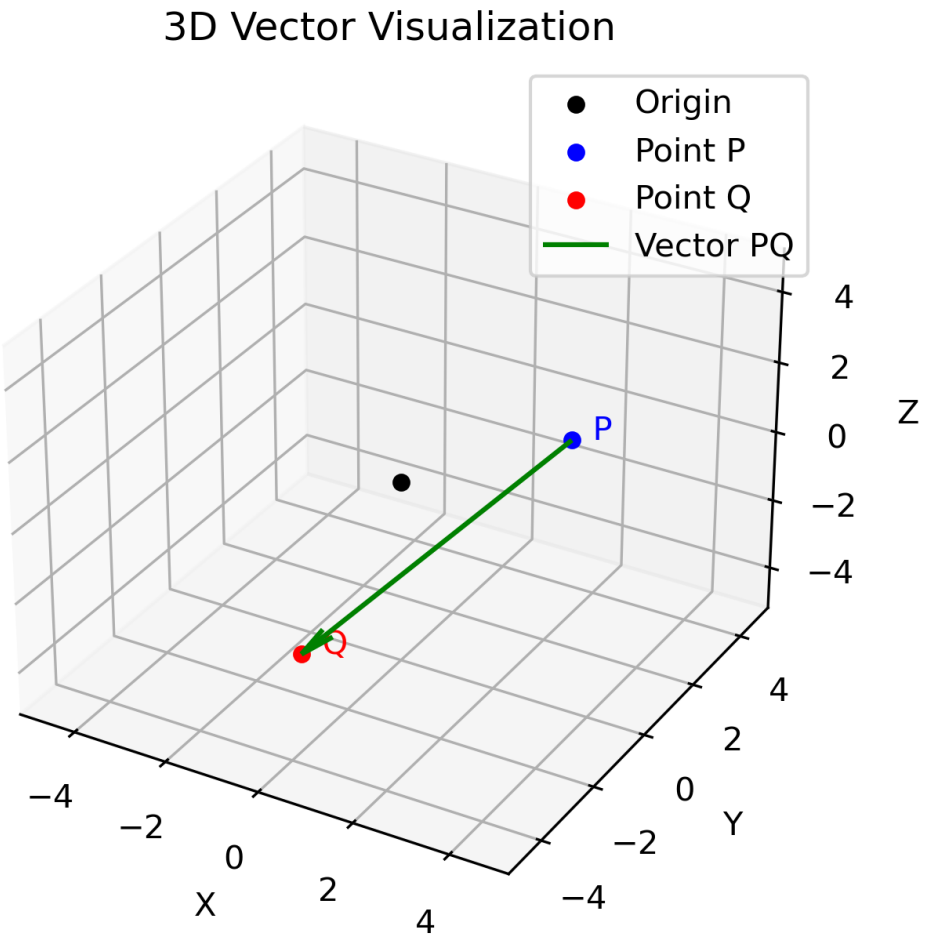
Table: Variables Used

The vector joining **P** and **Q** = **Q** - **P**

$$\implies \mathbf{Q} - \mathbf{P} = \begin{pmatrix} -1 \\ -2 \\ -4 \end{pmatrix} - \begin{pmatrix} 2 \\ 3 \\ 0 \end{pmatrix} = \begin{pmatrix} -3 \\ -5 \\ -4 \end{pmatrix}$$

$$\implies \text{The desired vector is } \begin{pmatrix} -3 \\ -5 \\ -4 \end{pmatrix}$$

Plot


3D Vector Visualization
Origin
Point P
Point Q
Vector PQ
P
Q
X
Y
Z
−4
−2
0
2
4

Figure

C Code: code.c

```c
#include <stdio.h>

int main() {
    FILE *fp;

    // Coordinates of P and Q
    int Px = 2, Py = 3, Pz = 0;
    int Qx = -1, Qy = -2, Qz = -4;

    // Vector from P to Q: Q - P
    int Vx = Qx - Px;
    int Vy = Qy - Py;
    int Vz = Qz - Pz;

    // Open file for writing
    fp = fopen("vector.dat", "w");
    if (fp == NULL) {
        printf("Error opening file!\n");
        return 1;
    }

    // Write vector to file
    fprintf(fp, "Vector from P(%d,%d,%d) to Q(%d,%d,%d):\n", Px, Py, Pz, Qx, Qy, Qz);
    fprintf(fp, "Vector PQ = (%d, %d, %d)\n", Vx, Vy, Vz);

    // Close file
    fclose(fp);

    printf("Vector successfully written to vector.dat\n");
    return 0;
}
```

Python: call.py

```
import subprocess
import os

# Compile the C code
compile_process = subprocess.run(["gcc", "code.c", "-o", "code.out"])

# Check if compilation was successful
if compile_process.returncode == 0:
    print("Compilation␣successful.␣Running␣the␣program...\n")

    # Run the compiled program
    run_process = subprocess.run(["./code.out"])
else:
    print("Compilation␣failed.")
```

Python: plot.py

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

# Read vector data from file
with open('vector.dat', 'r') as file:
    lines = file.readlines()

# Extract coordinates from the file
line1 = lines[0]
P_start = line1.split("P(")[1].split(")")[0]
Q_end = line1.split("Q(")[1].split(")")[0]

P = np.array(list(map(int, P_start.split(','))))
Q = np.array(list(map(int, Q_end.split(','))))
PQ = Q - P # Vector from P to Q

# Set up the 3D plot
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

# Plot origin
ax.scatter(0, 0, 0, color='black', label='Origin')

# Plot points
ax.scatter(*P, color='blue', label='Point␣P')
ax.scatter(*Q, color='red', label='Point␣Q')

# Plot vector PQ (arrow from P to Q)
ax.quiver(*P, *PQ, color='green', arrow_length_ratio=0.1, label='Vector␣PQ')

# Annotate points
ax.text(*P, '␣␣P', color='blue')
```

Python: plot.py

```
ax.text(*Q, '␣␣Q', color='red')

# Set limits
max_range = np.max(np.abs([P, Q])) + 1
ax.set_xlim([-max_range, max_range])
ax.set_ylim([-max_range, max_range])
ax.set_zlim([-max_range, max_range])

# Labels and legend
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.set_title('3D␣Vector␣Visualization')
ax.legend()

# Save the figure as an image (e.g., PNG)
plt.savefig('vector_plot.png', dpi=300) # Change filename/format if needed

print("Plot␣saved␣as␣vector_plot.png")
```