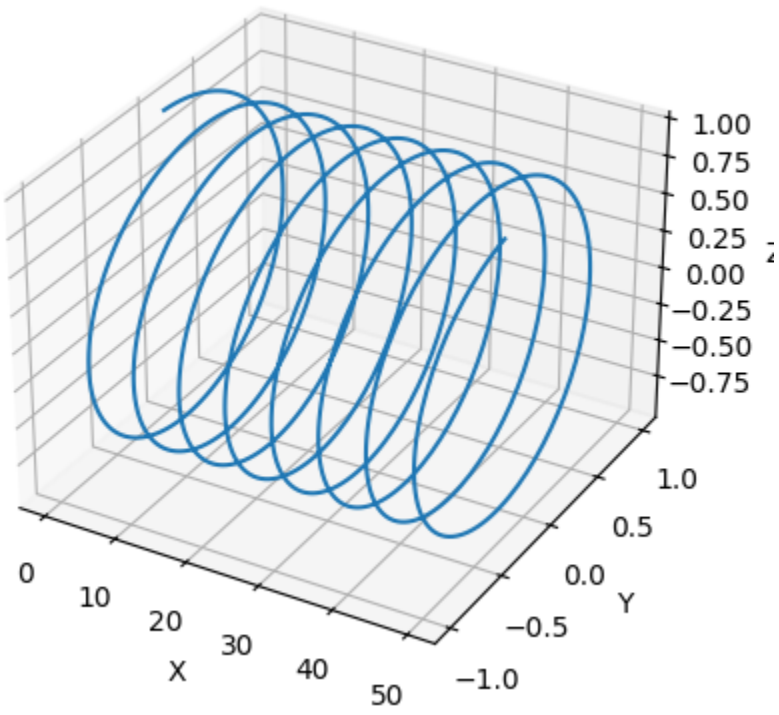
Joshua Kyle K. Entrata

4CSC

In [23]:
```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d.art3d import Poly3DCollection
import math
```

1. Create a 3D line plot that shows the path of a sine wave in 3D space.

In [6]:
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

x = np.linspace(0, 50, 1000)
y = np.sin(x)
z = np.cos(x)

ax.plot(x, y, z)
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')

plt.show()
```



2. Find the distance of the following: a. X to Y and (b) X to Z where X(2, 3, 6), Y(7, 8, 9), and Z (2, 9, 4).

In [7]:
```
def get_distance(p1, p2):
    p1 = np.array(p1)
    p2 = np.array(p2)

    distance = np.sqrt(np.sum((p2 - p1) ** 2))
    return distance
```

In [8]:
```
pt_x = (2, 3, 6)
pt_y = (7, 8, 9)
pt_z = (2, 9, 4)
```

In [11]:
```
xy = get_distance(pt_x, pt_y)
xz = get_distance(pt_x, pt_z)

print(f'Distance between X(2, 3, 6) and Y(7, 8, 9): {xy}')
print(f'Distance between X(2, 3, 6) and Z(2, 9, 4): {xz}')
```

```
Distance between X(2, 3, 6) and Y(7, 8, 9): 7.681145747868608
Distance between X(2, 3, 6) and Z(2, 9, 4): 6.324555320336759
```

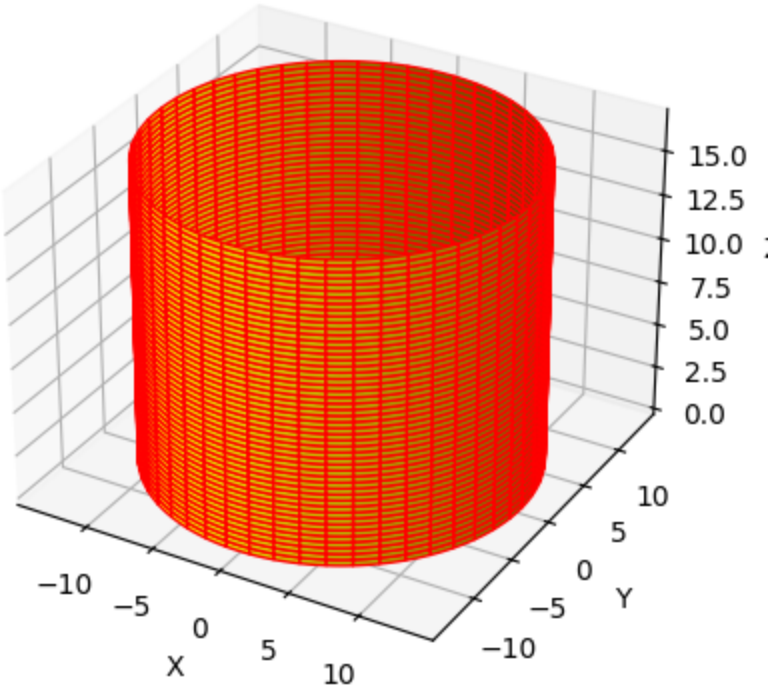
3. Draw the following in Python and compute for the volume of each 3D image (all edge color must be red).

a. Cylinder whose radius = 13.3, height = 17 and number of points/resolution = 450 (yellow)

In [24]:
```
radius_cylinder = 13.3
height_cylinder = 17
num_points_cylinder = 450
```

In [25]:
```
def draw_cylinder(radius, height, num_points, color, edge_color):
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')

    theta = np.linspace(0, 2*np.pi, num_points)
    z = np.linspace(0, height, num_points)
    theta, Z = np.meshgrid(theta, z)

    X = radius * np.cos(theta)
    Y = radius * np.sin(theta)

    ax.plot_surface(X, Y, Z, color=color, edgecolor=edge_color)

    ax.set_xlabel('X')
    ax.set_ylabel('Y')
    ax.set_zlabel('Z')

    plt.show()
```

In [26]:
```
draw_cylinder(radius_cylinder, height_cylinder, num_points_cylinder, color='yellow'
```



In [27]:
```
def volume_cylinder(radius, height):
    return math.pi * radius**2 * height

print("Volume of Cylinder:", volume_cylinder(radius_cylinder, height_cylinder))
```

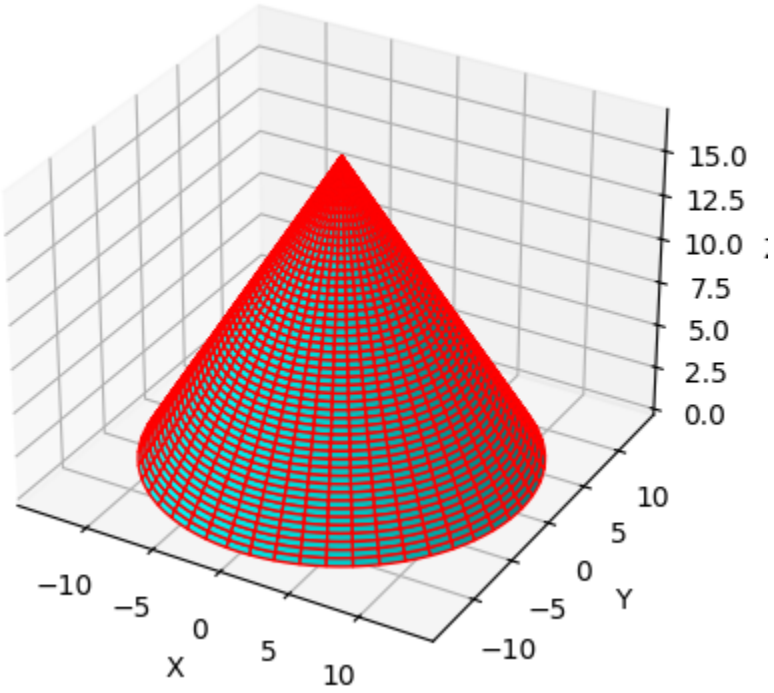
```
Volume of Cylinder: 9447.177516389474
```

b. Cone whose radius = 13.3, height = 17 and number of points/resolution = 450 (cyan)

In [28]:
```
radius_cone = 13.3
height_cone = 17
num_points_cone = 450
```

In [29]:
```
def draw_cone(radius, height, num_points, color, edge_color):
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')

    theta = np.linspace(0, 2*np.pi, num_points)
    z = np.linspace(0, height, num_points)
    theta, Z = np.meshgrid(theta, z)

    X = radius * (1 - Z/height) * np.cos(theta)
    Y = radius * (1 - Z/height) * np.sin(theta)

    ax.plot_surface(X, Y, Z, color=color, edgecolor=edge_color)

    ax.set_xlabel('X')
    ax.set_ylabel('Y')
    ax.set_zlabel('Z')

    plt.show()
```

In [30]:
```
draw_cone(radius_cone, height_cone, num_points_cone, color='cyan', edge_color='red'
```



In [31]:
```
def volume_cone(radius, height):
    return (1/3) * math.pi * radius**2 * height

print("Volume of Cone:", volume_cone(radius_cone, height_cone))
```

```
Volume of Cone: 3149.059172129825
```

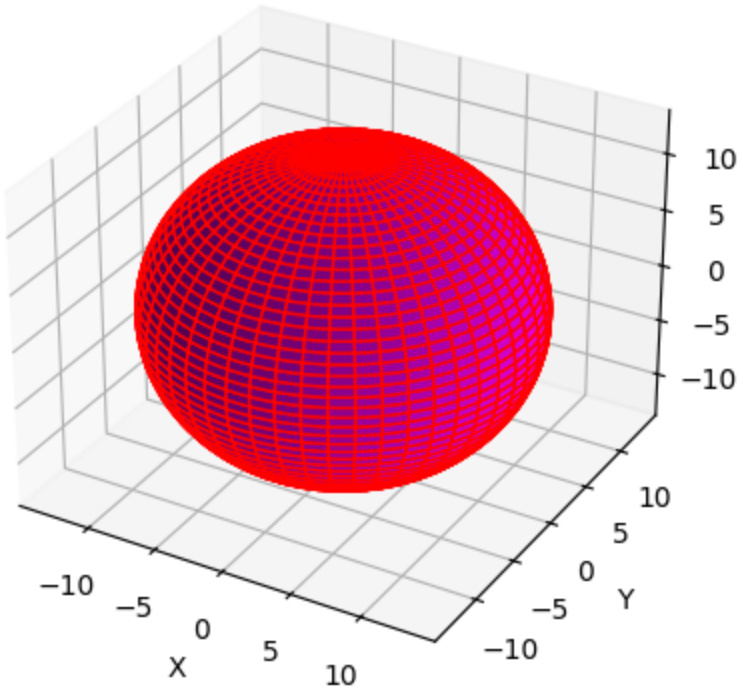
c. Sphere whose radius = 13.3 and number of points/resolution = 450 (magenta)

In [32]:
```
radius_sphere = 13.3
num_points_sphere = 450
```

In [33]:
```
def draw_sphere(radius, num_points, color, edge_color):
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')

    theta = np.linspace(0, 2 * np.pi, num_points)
    phi = np.linspace(0, np.pi, num_points)
    theta, phi = np.meshgrid(theta, phi)

    X = radius * np.sin(phi) * np.cos(theta)
    Y = radius * np.sin(phi) * np.sin(theta)
    Z = radius * np.cos(phi)

    ax.plot_surface(X, Y, Z, color=color, edgecolor=edge_color)

    ax.set_xlabel('X')
    ax.set_ylabel('Y')
    ax.set_zlabel('Z')
```

```
    plt.show()
```

In [34]:
```
draw_sphere(radius_sphere, num_points_sphere, color='magenta', edge_color='red')
```



In [35]:
```
def volume_sphere(radius):
    return (4/3) * math.pi * radius**3

print("Volume of Sphere:", volume_sphere(radius_sphere))
```

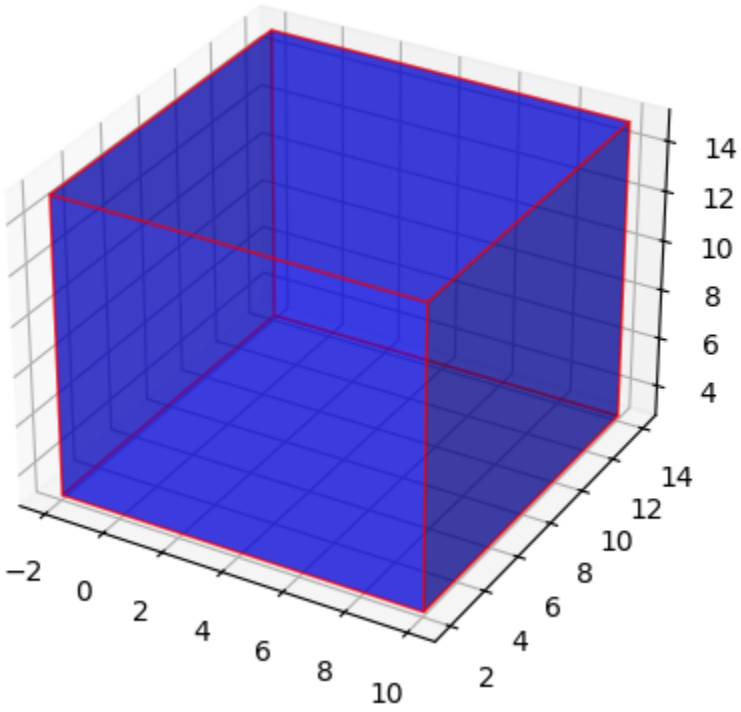
```
Volume of Sphere: 9854.70282101804
```

d. Cube whose initial coordinate located at (4, 8, 9) and the edge is 12. (blue)

In [36]:
```
coords_cube = (4, 8, 9)
edge_cube = 12
```

In [53]:
```
def draw_cube(coords, edge, color, edge_color):
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')

    r = [-edge / 2, edge / 2]
    X, Y = np.meshgrid(r, r)

    ax.plot_surface(X + coords[0], Y + coords[1], r[0] * np.ones_like(X) + coords[2
    ax.plot_surface(X + coords[0], Y + coords[1], r[1] * np.ones_like(X) + coords[2
    ax.plot_surface(X + coords[0], r[0] * np.ones_like(Y) + coords[1], Y + coords[2
    ax.plot_surface(X + coords[0], r[1] * np.ones_like(Y) + coords[1], Y + coords[2
    ax.plot_surface(r[0] * np.ones_like(X) + coords[0], X + coords[1], Y + coords[2
    ax.plot_surface(r[1] * np.ones_like(X) + coords[0], X + coords[1], Y + coords[2

    plt.show()
```

In [38]:
```
draw_cube(coords_cube, edge_cube, color='blue', edge_color='red')
```



In [40]:
```
def volume_cube(edge):
    return edge ** 3

print("Volume of Cube:", volume_cube(edge_cube))
```

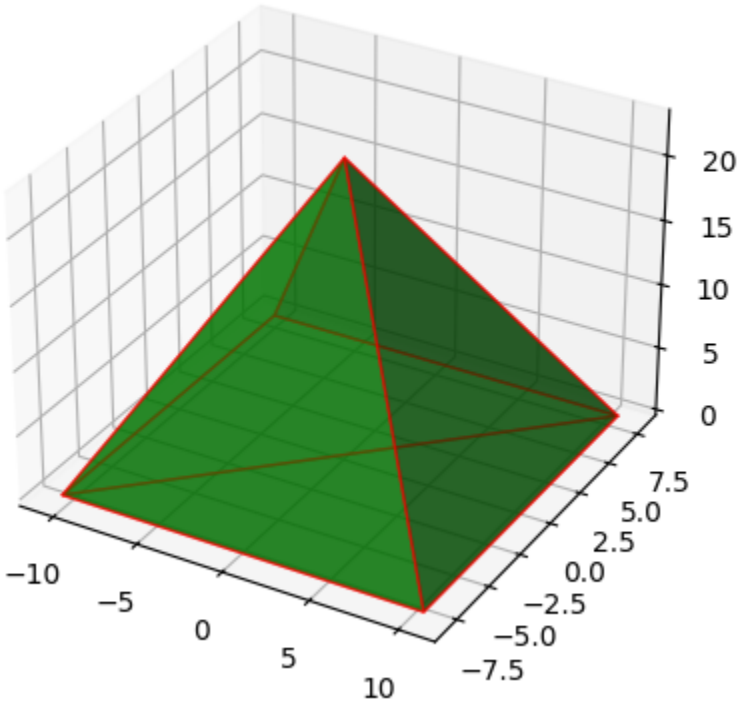
```
Volume of Cube: 1728
```

e. Pyramid whose base is rectangle (L = 21 and W = 16) and height is 23 units. (green)

In [43]:
```
length_pyramid = 21
width_pyramid = 16
height_pyramid = 23
```

In [41]:
```
def draw_pyramid(length, width, height, color, edge_color):
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')

    v = np.array([
        [-length/2, -width/2, 0],
        [length/2, -width/2, 0],
        [length/2, width/2, 0],
        [-length/2, width/2, 0],
        [0, 0, height]
    ])

    ax.plot_trisurf(v[[0, 1, 2], 0], v[[0, 1, 2], 1], v[[0, 1, 2], 2], color=color,
    ax.plot_trisurf(v[[0, 2, 3], 0], v[[0, 2, 3], 1], v[[0, 2, 3], 2], color=color,
    ax.plot_trisurf(v[[0, 1, 4], 0], v[[0, 1, 4], 1], v[[0, 1, 4], 2], color=color,
    ax.plot_trisurf(v[[1, 2, 4], 0], v[[1, 2, 4], 1], v[[1, 2, 4], 2], color=color,
    ax.plot_trisurf(v[[2, 3, 4], 0], v[[2, 3, 4], 1], v[[2, 3, 4], 2], color=color,
    ax.plot_trisurf(v[[3, 0, 4], 0], v[[3, 0, 4], 1], v[[3, 0, 4], 2], color=color,

    plt.show()
```

In [44]:
```
draw_pyramid(length_pyramid, width_pyramid, height_pyramid, color='green', edge_col
```



In [45]:
```
def volume_pyramid(length, width, height):
      return (1/3) * length * width *  height

print("Volume of Pyramid:", volume_pyramid(length_pyramid, width_pyramid, height_py
```

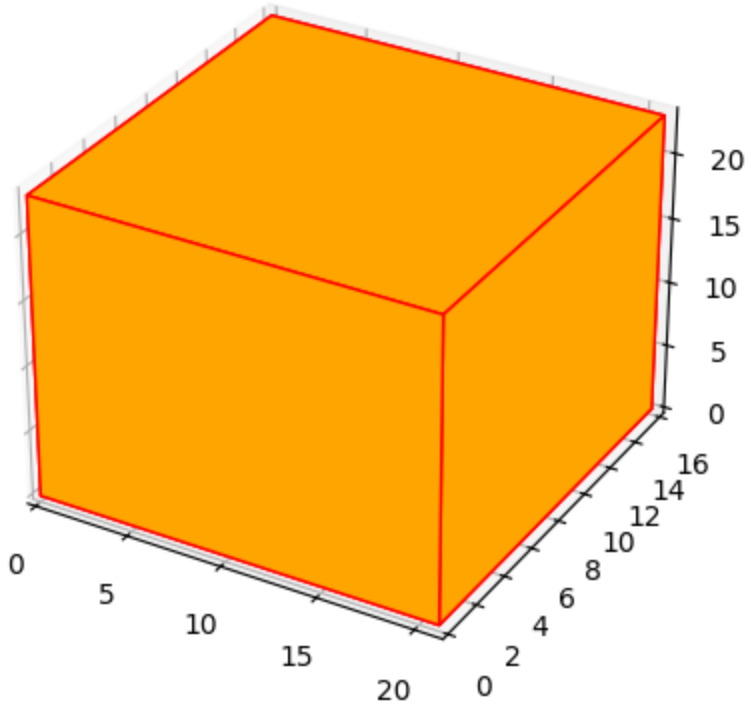
```
Volume of Pyramid: 2576.0
```

f. Prism whose base is rectangle (L = 21 and W = 16) and height is 23 units. (orange)

In [46]:
```
length_prism = 21
width_prism = 16
height_prism = 23
```

In [50]:
```
def draw_prism(length, width, height, color, edge_color):
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')

    v = np.array([
        [0, 0, 0],
        [length, 0, 0],
        [length, width, 0],
        [0, width, 0],
        [0, 0, height],
        [length, 0, height],
        [length, width, height],
        [0, width, height]
    ])

    faces = [
        [v[0], v[1], v[2], v[3]],
        [v[4], v[5], v[6], v[7]],
        [v[0], v[1], v[5], v[4]],
        [v[2], v[3], v[7], v[6]],
        [v[1], v[2], v[6], v[5]],
        [v[4], v[7], v[3], v[0]]
    ]

    ax.add_collection3d(Poly3DCollection(faces, facecolors=color, edgecolors=edge_c

    ax.set_xlim([0, length])
    ax.set_ylim([0, width])
    ax.set_zlim([0, height])

    plt.show()
```

In [51]:
```
draw_prism(length_prism, width_prism, height_prism, color='orange', edge_color='red
```



In [52]:
```
def volume_prism(length, width, height):
      return length * width * height

print("Volume of Prism:", volume_prism(length_prism, width_prism, height_prism))
```

```
Volume of Prism: 7728
```