LINEAR ALGEBRA IN BIOINFORMATICS

FIBRONIGON:

CODE:

(PDBDownloader.py):

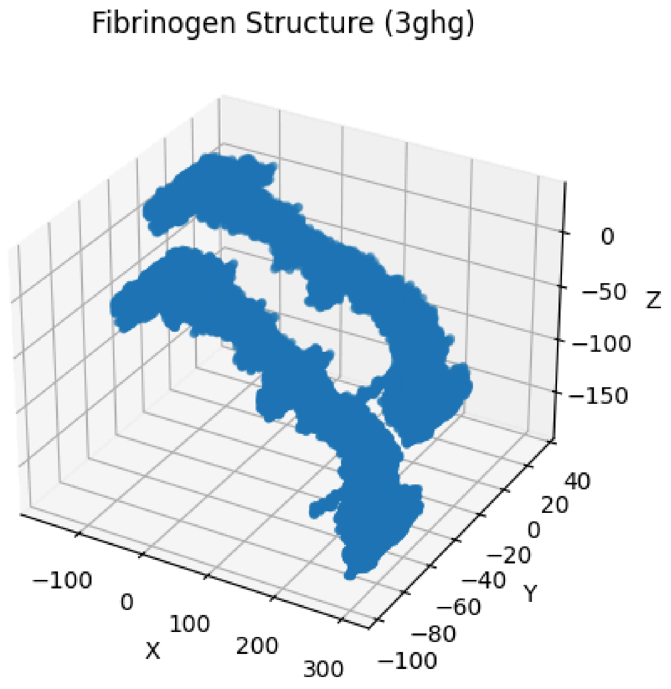
```
import os
import requests
from Bio.PDB import PDBParser

class PDBDownloader:
    def __init__(self, download_path="PDB_files"):
        self.download_path = download_path

    def download_pdb(self, pdb_id):
        try:
            if not os.path.exists(self.download_path):
                os.makedirs(self.download_path)

            pdb_url = f"https://files.rcsb.org/download/{pdb_id}.pdb"

            response = requests.get(pdb_url)
            if response.status_code == 200:
                pdb_filename = f"{pdb_id.lower()}.pdb"
                pdb_path = os.path.join(self.download_path, pdb_filename)

                with open(pdb_path, 'wb') as pdb_file:
                    pdb_file.write(response.content)

                print(f"Successfully downloaded {pdb_id} to {pdb_path}")

                header = self.extract_header(pdb_path)
                print("Header:")
                print(header)

            else:
                print(f"Failed to download {pdb_id}. Status code: {response.status_code}")
        except Exception as e:
            print(f"Error: {e}")

    def extract_header(self, pdb_path):
        parser = PDBParser(QUIET=True)
        structure = parser.get_structure('temp', pdb_path)
        header_lines = structure.header.get('head', [])
        header_str = ''.join(header_lines)
        return header_str

if __name__ == "__main__":
    pdb_downloader = PDBDownloader()
    pdb_id_to_download = '3ghg'
    pdb_downloader.download_pdb(pdb_id_to_download)
```

(Code.py):

```
from Bio import PDB
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

def extract_pdb_header(pdb_path):
    try:
        parser = PDB.PDBParser(QUIET=True)
        structure = parser.get_structure('temp', pdb_path)
        header_lines = structure.header.get('head', [])
        header_str = ''.join(header_lines)
        return header_str
    except Exception as e:
        print(f"Error: {e}")
        return None

pdb_id = "3ghg"
pdb_filename = f"path of the pdb file downloaded in my computer"
structure = PDB.PDBParser(QUIET=True).get_structure(pdb_id, pdb_filename)

atom_coords = []
```

```
for model in structure:
    for chain in model:
        for residue in chain:
            for atom in residue:
                atom_coords.append(atom.get_coord())

atom_coords_array = np.array(atom_coords)

pdb_header = extract_pdb_header(pdb_filename)
print(pdb_header)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(atom_coords_array[:, 0], atom_coords_array[:, 1], atom_coords_array[:, 2],
marker='o', s=10)
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.set_title(f'{pdb_header} ({pdb_id})')

plt.show()
```

OUTPUT:



HYDROLASE:

CODE:

(PDBDownloader.py):

```
import os
import requests
from Bio.PDB import PDBParser

class PDBDownloader:
    def __init__(self, download_path="PDB_files"):
        self.download_path = download_path

    def download_pdb(self, pdb_id):
        try:
            if not os.path.exists(self.download_path):
                os.makedirs(self.download_path)

            pdb_url = f"https://files.rcsb.org/download/{pdb_id}.pdb"

            response = requests.get(pdb_url)
            if response.status_code == 200:
                pdb_filename = f"{pdb_id.lower()}.pdb"
                pdb_path = os.path.join(self.download_path, pdb_filename)

                with open(pdb_path, 'wb') as pdb_file:
                    pdb_file.write(response.content)

                print(f"Successfully downloaded {pdb_id} to {pdb_path}")

                header = self.extract_header(pdb_path)
                print("Header:")
                print(header)

            else:
                print(f"Failed to download {pdb_id}. Status code: {response.status_code}")
        except Exception as e:
            print(f"Error: {e}")

    def extract_header(self, pdb_path):
        parser = PDBParser(QUIET=True)
        structure = parser.get_structure('temp', pdb_path)
        header_lines = structure.header.get('head', [])
        header_str = ''.join(header_lines)
        return header_str

if __name__ == "__main__":
    pdb_downloader = PDBDownloader()
    pdb_id_to_download = '1g0a'
    pdb_downloader.download_pdb(pdb_id_to_download)
```

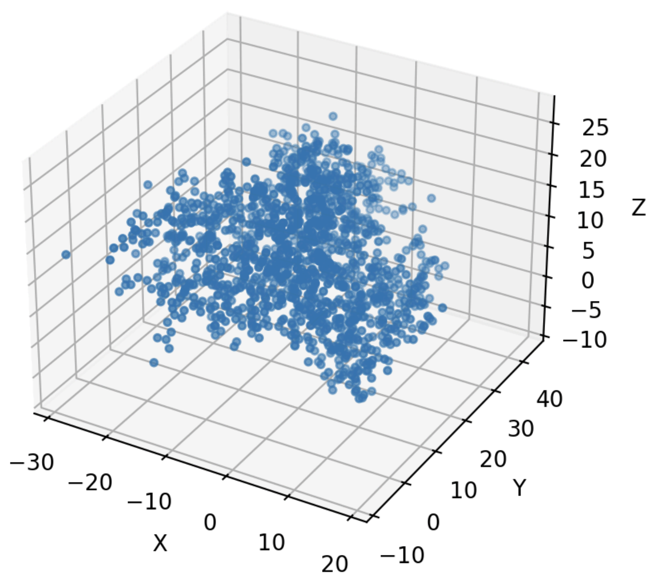
(Code.py):

```
from Bio import PDB
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

def extract_pdb_header(pdb_path):
    try:
        parser = PDB.PDBParser(QUIET=True)
        structure = parser.get_structure('temp', pdb_path)
        header_lines = structure.header.get('head', [])
        header_str = ''.join(header_lines)
        return header_str
    except Exception as e:
        print(f"Error: {e}")
        return None

pdb_id = "1g0a"
pdb_filename = f"path of the pdb file downloaded in my computer"
structure = PDB.PDBParser(QUIET=True).get_structure(pdb_id, pdb_filename)

atom_coords = []
```

```
for model in structure:
    for chain in model:
        for residue in chain:
            for atom in residue:
                atom_coords.append(atom.get_coord())

atom_coords_array = np.array(atom_coords)

pdb_header = extract_pdb_header(pdb_filename)
print(pdb_header)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(atom_coords_array[:, 0], atom_coords_array[:, 1], atom_coords_array[:, 2],
marker='o', s=10)
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.set_title(f'{pdb_header} ({pdb_id})')

plt.show()
```

OUTPUT:

hydrolase(endoribonuclease) (1GOA)



LYSOZYME:

CODE:

(PDBDownloader.py):

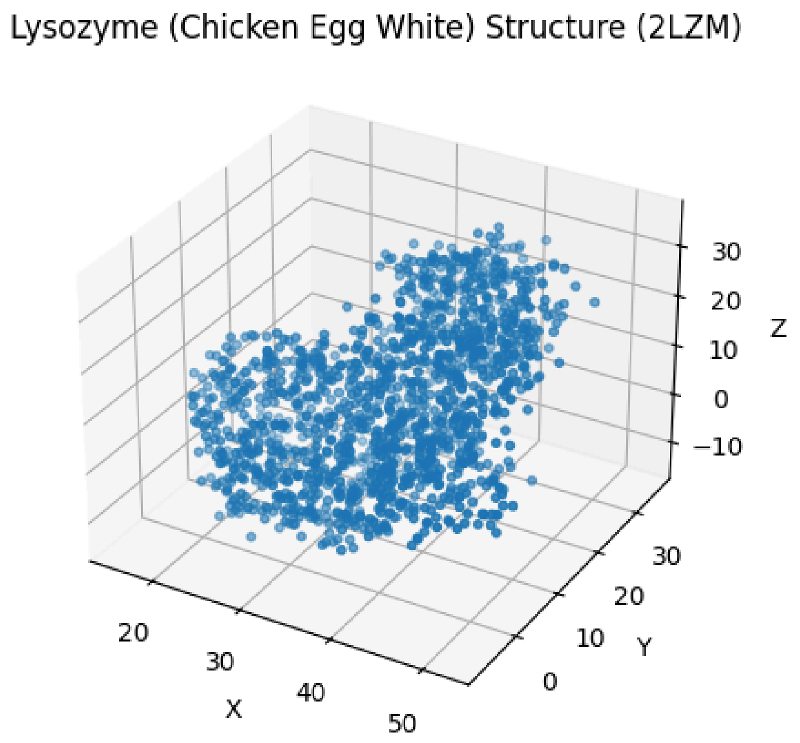
```
import os
import requests
from Bio.PDB import PDBParser

class PDBDownloader:
    def __init__(self, download_path="PDB_files"):
        self.download_path = download_path

    def download_pdb(self, pdb_id):
        try:
            if not os.path.exists(self.download_path):
                os.makedirs(self.download_path)

            pdb_url = f"https://files.rcsb.org/download/{pdb_id}.pdb"

            response = requests.get(pdb_url)
            if response.status_code == 200:
                pdb_filename = f"{pdb_id.lower()}.pdb"
                pdb_path = os.path.join(self.download_path, pdb_filename)

                with open(pdb_path, 'wb') as pdb_file:
                    pdb_file.write(response.content)

                print(f"Successfully downloaded {pdb_id} to {pdb_path}")

                header = self.extract_header(pdb_path)
                print("Header:")
                print(header)

            else:
                print(f"Failed to download {pdb_id}. Status code: {response.status_code}")
        except Exception as e:
            print(f"Error: {e}")

    def extract_header(self, pdb_path):
        parser = PDBParser(QUIET=True)
        structure = parser.get_structure('temp', pdb_path)
        header_lines = structure.header.get('head', [])
        header_str = ''.join(header_lines)
        return header_str

if __name__ == "__main__":
    pdb_downloader = PDBDownloader()
    pdb_id_to_download = '2LZM'
    pdb_downloader.download_pdb(pdb_id_to_download)
```

(Code.py):

```
from Bio import PDB
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

def extract_pdb_header(pdb_path):
    try:
        parser = PDB.PDBParser(QUIET=True)
        structure = parser.get_structure('temp', pdb_path)
        header_lines = structure.header.get('head', [])
        header_str = ''.join(header_lines)
        return header_str
    except Exception as e:
        print(f"Error: {e}")
        return None

pdb_id = "2LZM"
pdb_filename = f"path of the pdb file downloaded in my computer"
structure = PDB.PDBParser(QUIET=True).get_structure(pdb_id, pdb_filename)

atom_coords = []
```

```
for model in structure:
    for chain in model:
        for residue in chain:
            for atom in residue:
                atom_coords.append(atom.get_coord())

atom_coords_array = np.array(atom_coords)

pdb_header = extract_pdb_header(pdb_filename)
print(pdb_header)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(atom_coords_array[:, 0], atom_coords_array[:, 1], atom_coords_array[:, 2],
marker='o', s=10)
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.set_title(f'{pdb_header} ({pdb_id})')

plt.show()
```

OUTPUT:



DNA:

CODE:

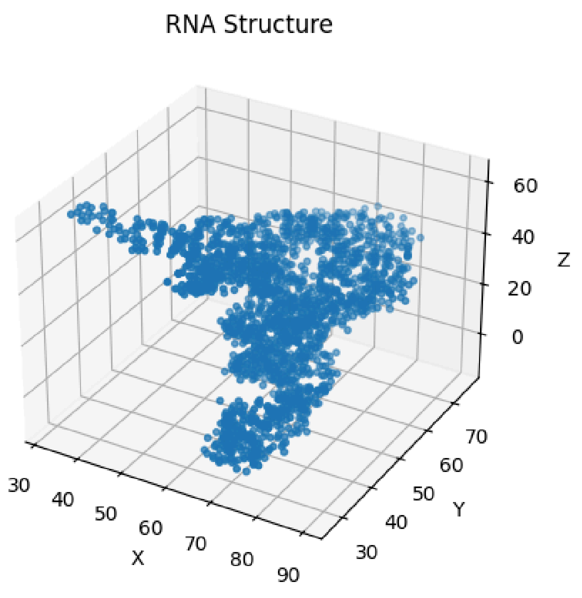
(PDBDownloader.py):

```
import os
import requests
from Bio.PDB import PDBParser

class PDBDownloader:
    def __init__(self, download_path="PDB_files"):
        self.download_path = download_path

    def download_pdb(self, pdb_id):
        try:
            if not os.path.exists(self.download_path):
                os.makedirs(self.download_path)

            pdb_url = f"https://files.rcsb.org/download/{pdb_id}.pdb"

            response = requests.get(pdb_url)
            if response.status_code == 200:
                pdb_filename = f"{pdb_id.lower()}.pdb"
                pdb_path = os.path.join(self.download_path, pdb_filename)

                with open(pdb_path, 'wb') as pdb_file:
                    pdb_file.write(response.content)

                print(f"Successfully downloaded {pdb_id} to {pdb_path}")

                header = self.extract_header(pdb_path)
                print("Header:")
                print(header)

            else:
                print(f"Failed to download {pdb_id}. Status code: {response.status_code}")
        except Exception as e:
            print(f"Error: {e}")

    def extract_header(self, pdb_path):
        parser = PDBParser(QUIET=True)
        structure = parser.get_structure('temp', pdb_path)
        header_lines = structure.header.get('head', [])
        header_str = ''.join(header_lines)
        return header_str

if __name__ == "__main__":
    pdb_downloader = PDBDownloader()
    pdb_id_to_download = '1EHZ'
    pdb_downloader.download_pdb(pdb_id_to_download)
```

(Code.py):

```
from Bio import PDB
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

def extract_pdb_header(pdb_path):
    try:
        parser = PDB.PDBParser(QUIET=True)
        structure = parser.get_structure('temp', pdb_path)
        header_lines = structure.header.get('head', [])
        header_str = ''.join(header_lines)
        return header_str
    except Exception as e:
        print(f"Error: {e}")
        return None

pdb_id = "1EHZ"
pdb_filename = f"path of the pdb file downloaded in my computer"
structure = PDB.PDBParser(QUIET=True).get_structure(pdb_id, pdb_filename)

atom_coords = []
for model in structure:
    for chain in model:
        for residue in chain:
            for atom in residue:
                atom_coords.append(atom.get_coord())

atom_coords_array = np.array(atom_coords)

pdb_header = extract_pdb_header(pdb_filename)
print(pdb_header)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(atom_coords_array[:, 0], atom_coords_array[:, 1], atom_coords_array[:, 2],
marker='o', s=10)
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.set_title(f'{pdb_header} ({pdb_id})')

plt.show()
```

OUTPUT:


RNA Structure
Z
Y
X