Assignment #06- Python Pandas

Name: Soumyadeep Ganguly

Reg. No.: 24MDT0082

M.Sc. Data Science

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Q1. Read the data file NutAverage.xlsx into a DataFrame and perform the following tasks:

In [2]:
```
df = pd.read_excel('Pandas/NutAverage.xlsx')
df.head()
```

Out[2]:

	Day Count	NH4-N	NO2-N	NO3-N	TN
0	1	6.915879	2.885372	7.457832	35.834969
1	47	6.344965	2.852123	5.696753	36.359106
2	78	4.964745	2.090747	2.167375	40.719987
3	116	4.361492	2.301630	1.653266	24.931194
4	143	3.980372	1.419541	0.233538	36.234797

Find the null values, if any, in the data set and fill the null values with the method of your choice

In [3]:
```
df.isna().sum()
```

Out[3]:
```
Day Count    0
NH4-N        0
NO2-N        0
NO3-N        0
TN           0
dtype: int64
```

Add a column DIN (stands for Dissolved Inorganic Nitrogen) to this DataFrame, where DIN = NH4-N+NO2-N+NO3-N.

In [4]:
```
df.columns
```

```
Out[4]: Index(['Day Count', 'NH4-N', 'NO2-N', 'NO3-N', 'TN'], dtype='object')
```

```
In [5]: df['DIN'] = df['NH4-N'] + df['NO2-N'] + df['NO3-N']
        df
```

Out[5]:

	Day Count	NH4-N	NO2-N	NO3-N	TN	DIN
0	1	6.915879	2.885372	7.457832	35.834969	17.259083
1	47	6.344965	2.852123	5.696753	36.359106	14.893841
2	78	4.964745	2.090747	2.167375	40.719987	9.222866
3	116	4.361492	2.301630	1.653266	24.931194	8.316388
4	143	3.980372	1.419541	0.233538	36.234797	5.633451
5	181	4.814007	1.416273	0.185584	36.269086	6.415864
6	210	5.774826	2.250251	1.034297	27.557018	9.059374
7	236	4.439287	1.977844	0.482635	45.557639	6.899766
8	270	2.394753	2.289396	1.439277	28.102278	6.123425
9	298	1.956891	1.685278	0.831846	30.017141	4.474016
10	332	3.965562	2.330494	1.613106	31.488240	7.909163
11	364	4.521629	2.305632	1.670597	41.095589	8.497858

Add another column DON (Dissolved Organic Nitrogen) to this DataFrame, where DON = TN - DIN.

```
In [6]: df['DON'] = df['TN'] - df['DIN']
        df
```

Out[6]:

	Day Count	NH4-N	NO2-N	NO3-N	TN	DIN	DON
0	1	6.915879	2.885372	7.457832	35.834969	17.259083	18.575886
1	47	6.344965	2.852123	5.696753	36.359106	14.893841	21.465266
2	78	4.964745	2.090747	2.167375	40.719987	9.222866	31.497121
3	116	4.361492	2.301630	1.653266	24.931194	8.316388	16.614806
4	143	3.980372	1.419541	0.233538	36.234797	5.633451	30.601346
5	181	4.814007	1.416273	0.185584	36.269086	6.415864	29.853222
6	210	5.774826	2.250251	1.034297	27.557018	9.059374	18.497643
7	236	4.439287	1.977844	0.482635	45.557639	6.899766	38.657873
8	270	2.394753	2.289396	1.439277	28.102278	6.123425	21.978853
9	298	1.956891	1.685278	0.831846	30.017141	4.474016	25.543126
10	332	3.965562	2.330494	1.613106	31.488240	7.909163	23.579077
11	364	4.521629	2.305632	1.670597	41.095589	8.497858	32.597731

Add a row named Averages which contains the averages of each of NH4-N, NO2-N, NO3-N, TN, DIN, DON

In [7]:

```
df.loc['Average'] = [np.mean(df[i]) for i in df.columns]
df
```

Out[7]:

	Day Count	NH4-N	NO2-N	NO3-N	TN	DIN	DON
0	1.000000	6.915879	2.885372	7.457832	35.834969	17.259083	18.575886
1	47.000000	6.344965	2.852123	5.696753	36.359106	14.893841	21.465266
2	78.000000	4.964745	2.090747	2.167375	40.719987	9.222866	31.497121
3	116.000000	4.361492	2.301630	1.653266	24.931194	8.316388	16.614806
4	143.000000	3.980372	1.419541	0.233538	36.234797	5.633451	30.601346
5	181.000000	4.814007	1.416273	0.185584	36.269086	6.415864	29.853222
6	210.000000	5.774826	2.250251	1.034297	27.557018	9.059374	18.497643
7	236.000000	4.439287	1.977844	0.482635	45.557639	6.899766	38.657873
8	270.000000	2.394753	2.289396	1.439277	28.102278	6.123425	21.978853
9	298.000000	1.956891	1.685278	0.831846	30.017141	4.474016	25.543126
10	332.000000	3.965562	2.330494	1.613106	31.488240	7.909163	23.579077
11	364.000000	4.521629	2.305632	1.670597	41.095589	8.497858	32.597731
Average	189.666667	4.536201	2.150382	2.038842	34.513920	8.725425	25.788496

Describe characteristics of the DataFrame

In [8]:
```
df.describe()
```

Out[8]:

	Day Count	NH4-N	NO2-N	NO3-N	TN	DIN	DON
count	13.000000	13.000000	13.000000	13.000000	13.000000	13.000000	13.000000
mean	189.666667	4.536201	2.150382	2.038842	34.513920	8.725425	25.788496
std	110.867288	1.373646	0.453143	2.144153	5.956099	3.596171	6.530319
min	1.000000	1.956891	1.416273	0.185584	24.931194	4.474016	16.614806
25%	116.000000	3.980372	1.977844	0.831846	30.017141	6.415864	21.465266
50%	189.666667	4.521629	2.250251	1.613106	35.834969	8.316388	25.543126
75%	270.000000	4.964745	2.305632	2.038842	36.359106	9.059374	30.601346
max	364.000000	6.915879	2.885372	7.457832	45.557639	17.259083	38.657873

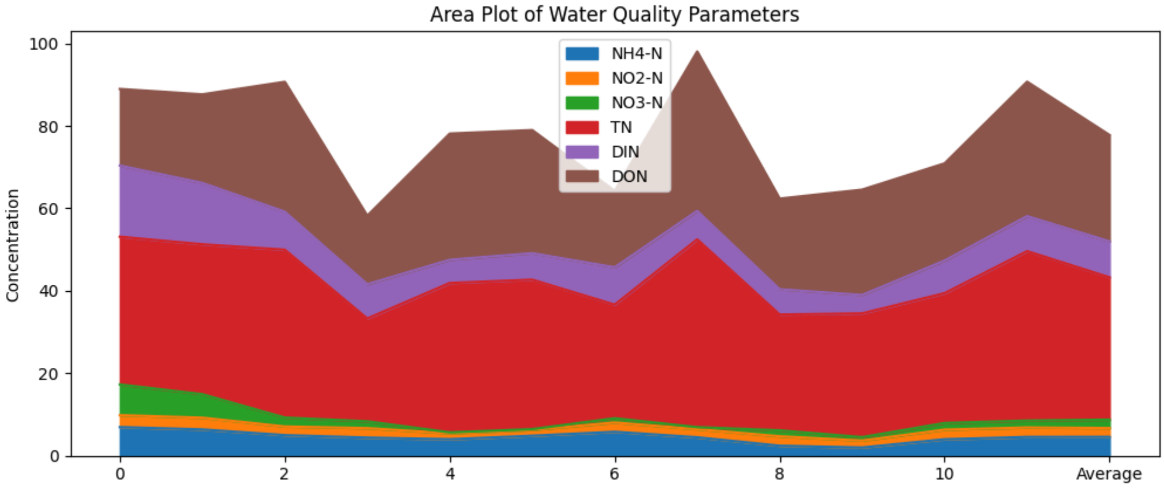
Plot all the data (except the Day Count column) using the area plot, line plot and box plot of DataFrame. (use Subplots where ever appropriate for better visualisation of the data)

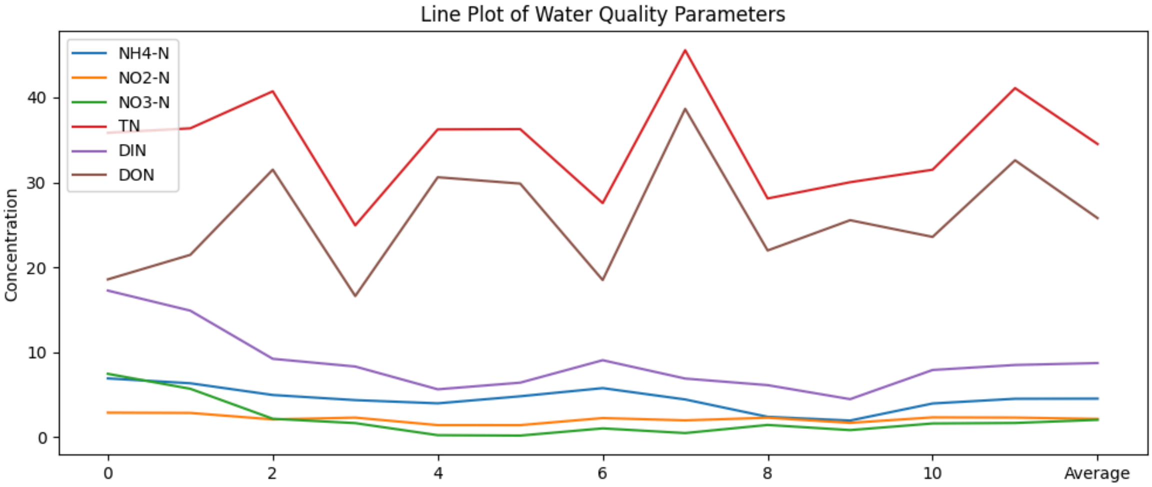
In [9]:
```
df.columns
```

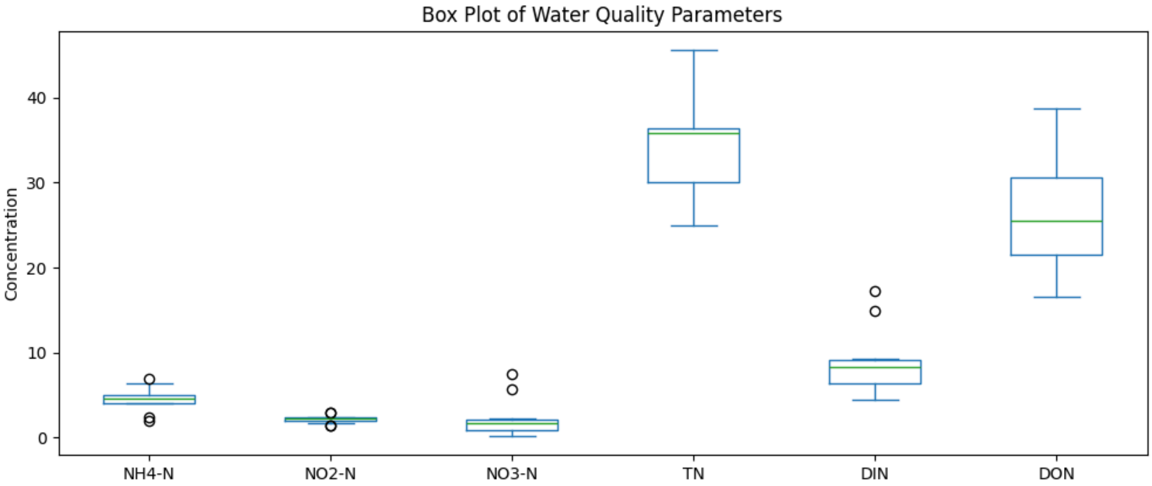
Out[9]:
```
Index(['Day Count', 'NH4-N', 'NO2-N', 'NO3-N', 'TN', 'DIN', 'DON'], dtype='obje
ct')
```

In [10]:
```
df2 = df[['NH4-N', 'NO2-N', 'NO3-N', 'TN', 'DIN', 'DON']]

fig, axs = plt.subplots(3, 1, figsize=(12, 16))

df2.plot.area(ax=axs[0])
axs[0].set_title('Area Plot of Water Quality Parameters')
axs[0].set_ylabel('Concentration')

df2.plot(ax=axs[1])
axs[1].set_title('Line Plot of Water Quality Parameters')
axs[1].set_ylabel('Concentration')

df2.plot.box(ax=axs[2])
axs[2].set_title('Box Plot of Water Quality Parameters')
axs[2].set_ylabel('Concentration')
```

Out[10]:
```
Text(0, 0.5, 'Concentration')
```

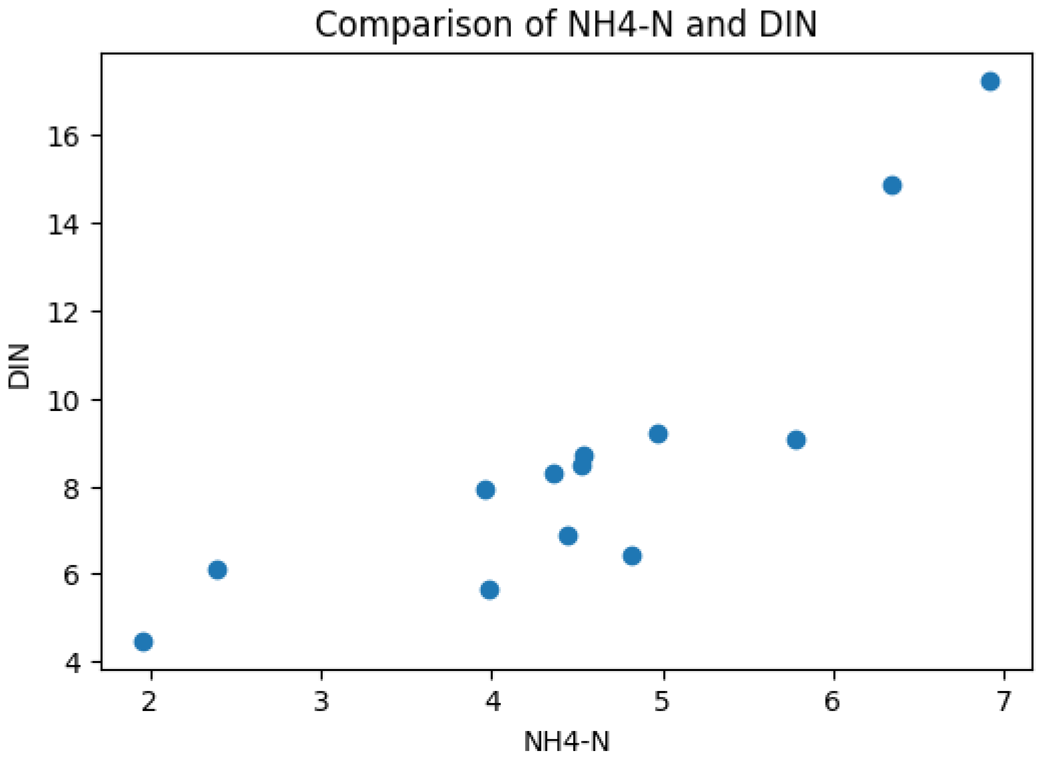






Compare the NH4-N vs DIN composition graphically using a scatter plot.

In [11]:
```
fig, ax = plt.subplots(figsize=(6,4))
ax.scatter(df['NH4-N'], df['DIN'])
ax.set_xlabel('NH4-N')
ax.set_ylabel('DIN')
ax.set_title('Comparison of NH4-N and DIN')
plt.show()
```



Q2. Read the data file PhythoBiomass.xlsx into a DataFrame and perform the following tasks:

In [12]:

```
df = pd.read_excel('Pandas/PythoBiomass.xlsx')
df.head()
```

Out[12]:

	Days	Cyanophyceans	Chlorophyceae	Total Biomass
0	1	0.554035	0.340955	4.775824
1	47	0.409126	0.446749	4.536462
2	78	0.606581	0.210896	4.131376
3	116	0.308334	1.301525	3.597625
4	143	0.828900	0.352965	2.867716

Find the null values, if any, in the data set and fill these null values with the method of your choice

In [13]:

```
df.isna().count()
```

Out[13]:

```
Days             12
Cyanophyceans    12
Chlorophyceae    12
Total Biomass    12
dtype: int64
```

In [14]:

```
df = df.fillna(df.mean())
df.isna().sum()
```

Out[14]:

```
Days              0
Cyanophyceans     0
Chlorophyceae     0
Total Biomass     0
dtype: int64
```

Add a column Others which list the biomass of other phytoplankton groups obtained by subtracting TotalBiomass with the sum of the biomass of Cyanophycean and Chlorophyceae.

In [15]:

```
df = df.rename(columns={'Total Biomass ':'Total Biomass'})
```

In [16]:

```
df.columns
```

Out[16]:

```
Index(['Days', 'Cyanophyceans', 'Chlorophyceae', 'Total Biomass'], dtype='objec
t')
```

In [17]:

```
df['Others'] = df['Total Biomass'] - (df['Cyanophyceans']-df['Chlorophyceae'])
df
```

Out[17]:

	Days	Cyanophyceans	Chlorophyceae	Total Biomass	Others
0	1	0.554035	0.340955	4.775824	4.562744
1	47	0.409126	0.446749	4.536462	4.574084
2	78	0.606581	0.210896	4.131376	3.735692
3	116	0.308334	1.301525	3.597625	4.590816
4	143	0.828900	0.352965	2.867716	2.391781
5	181	0.822262	1.327444	3.179547	3.684730
6	210	3.303263	1.168384	7.895237	5.760359
7	236	35.462698	0.739803	40.195265	5.472369
8	270	3.882161	0.537869	9.079613	5.735321
9	298	0.575795	0.399925	3.351554	3.175684
10	332	1.276101	1.322828	9.098784	9.145511
11	364	1.127914	0.344519	10.457813	9.674417

Describe the characteristics of the DataFrame.

In [18]:

```
df.describe()
```

Out[18]:

	Days	Cyanophyceans	Chlorophyceae	Total Biomass	Others
count	12.000000	12.000000	12.000000	12.000000	12.000000
mean	189.666667	4.096431	0.707822	8.597235	5.208626
std	115.797106	9.944160	0.443034	10.307117	2.212660
min	1.000000	0.308334	0.210896	2.867716	2.391781
25%	106.500000	0.570355	0.350853	3.536107	3.722951
50%	195.500000	0.825581	0.492309	4.656143	4.582450
75%	277.000000	1.782892	1.201670	9.084406	5.741581
max	364.000000	35.462698	1.327444	40.195265	9.674417

Plot the biomass composition of each group using a barh and kde plot.

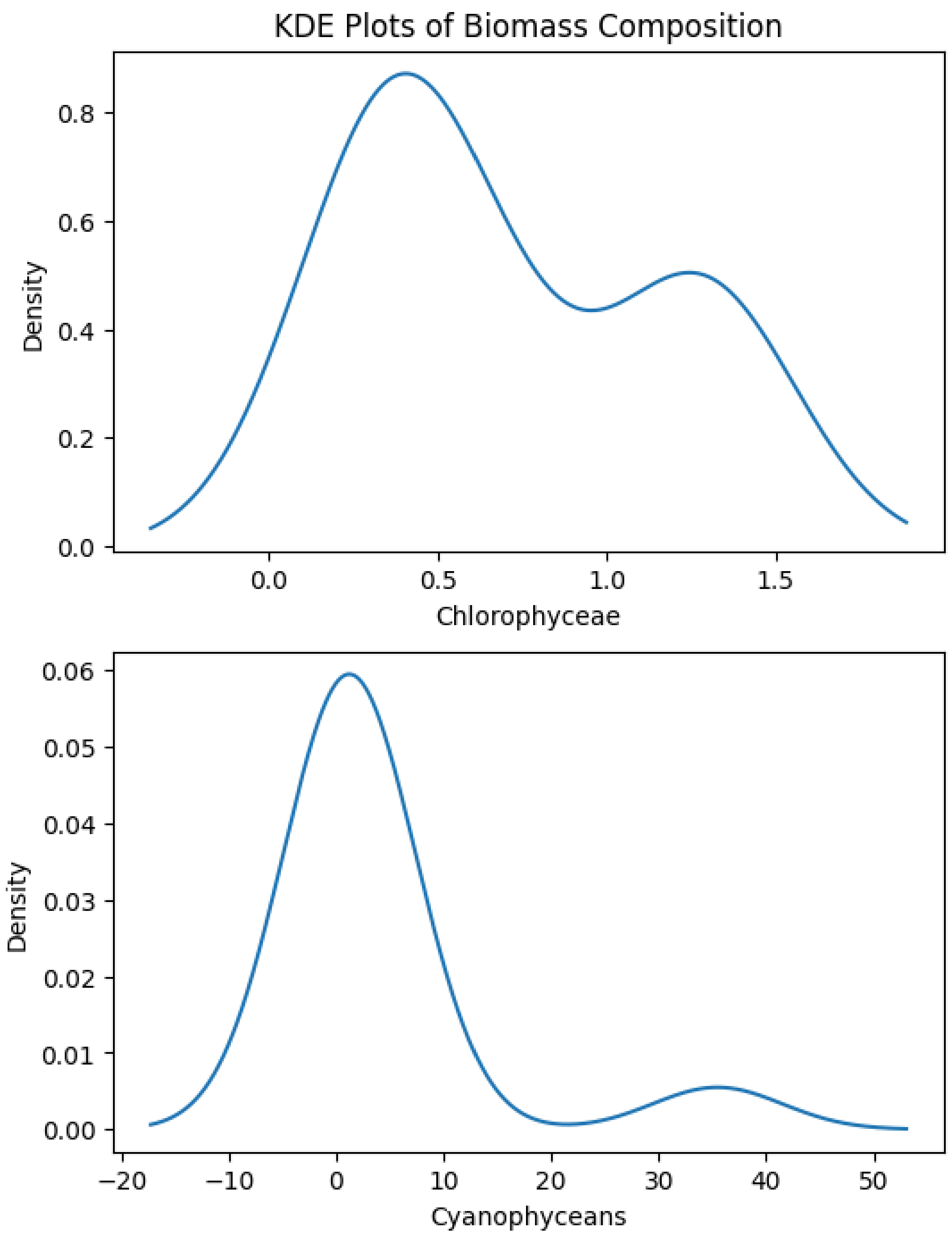
KDE Plots

In [19]:

```python
fig, axs = plt.subplots(2,1, figsize=(6, 8))
df['Chlorophyceae'].plot.kde(ax=axs[0])
axs[0].set_xlabel('Chlorophyceae')
df['Cyanophyceans'].plot.kde(ax=axs[1])
axs[1].set_xlabel('Cyanophyceans')
axs[0].set_title("KDE Plots of Biomass Composition")
```

Out[19]:

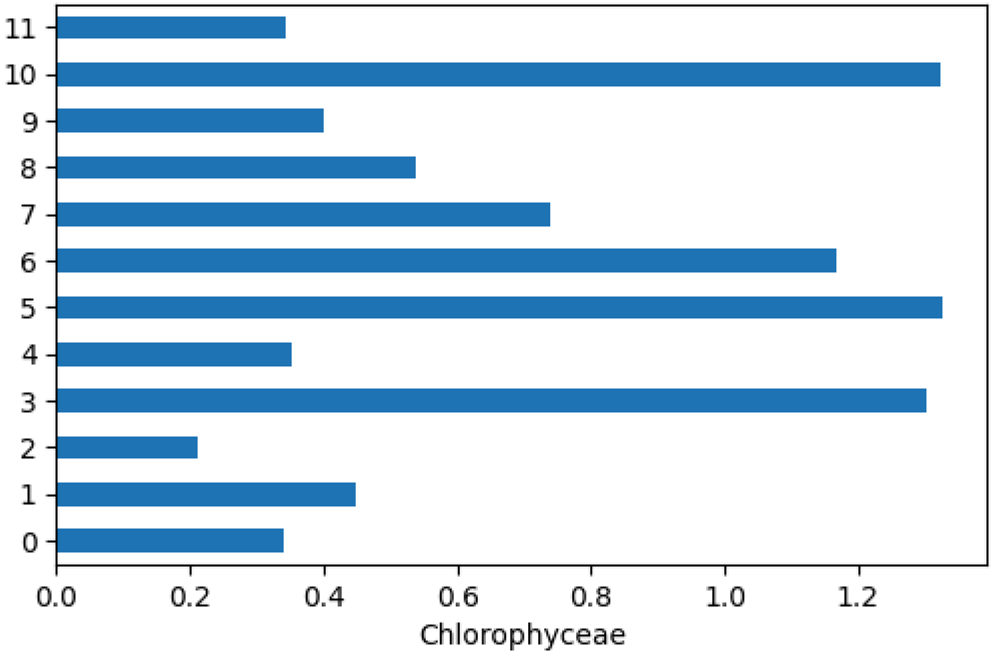
```
Text(0.5, 1.0, 'KDE Plots of Biomass Composition')
```

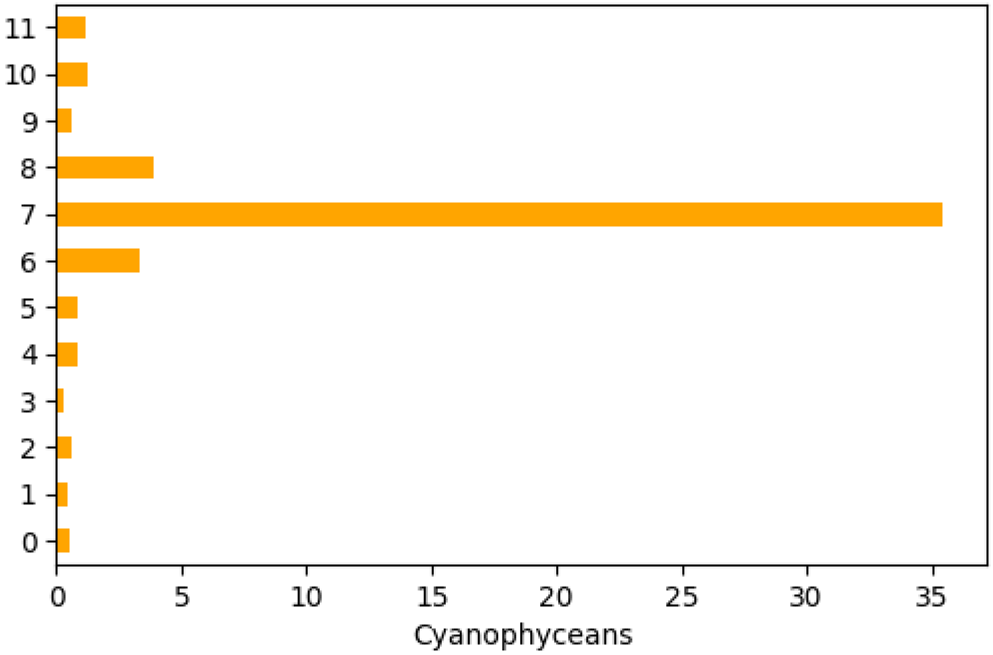


Barh Plots

```
In [20]: fig, axs = plt.subplots(2,1, figsize=(6, 8))
         df['Chlorophyceae'].plot.barh(ax=axs[0], x=np.arange(len(df['Chlorophyceae'])) )
         axs[0].set_xlabel('Chlorophyceae')
         df['Cyanophyceans'].plot.barh(ax=axs[1], x=np.arange(len(df['Cyanophyceans'])),
         axs[1].set_xlabel('Cyanophyceans')
```

```
Out[20]: Text(0.5, 0, 'Cyanophyceans')
```





Q3. Read the data file DOData.csv into a DataFrame and perform the following tasks:

In [21]:
```
df = pd.read_csv('Pandas/DOData.csv')
df2 = pd.read_excel('Pandas/NutAverage.xlsx')

df
```

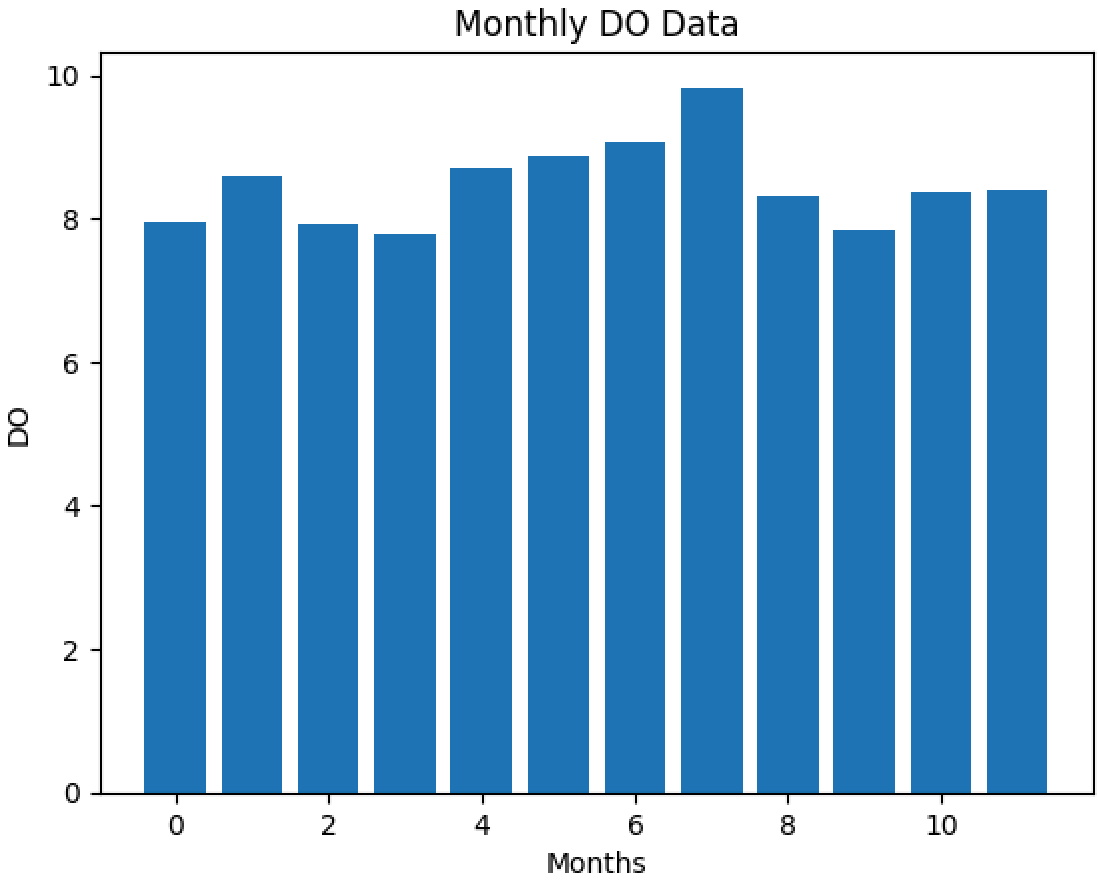
Out[21]:

	Days	DO
0	1	7.96
1	47	8.60
2	78	7.92
3	116	7.78
4	143	8.70
5	181	8.87
6	210	9.07
7	236	9.83
8	270	8.32
9	298	7.85
10	332	8.38
11	365	8.40

Plot the monthly DO data using a bar plot.

In [22]:

```
plt.bar(np.arange(len(df['Days'])), df['DO'])
plt.title('Monthly DO Data')
plt.xlabel('Months')
plt.ylabel('DO')
plt.show()
```



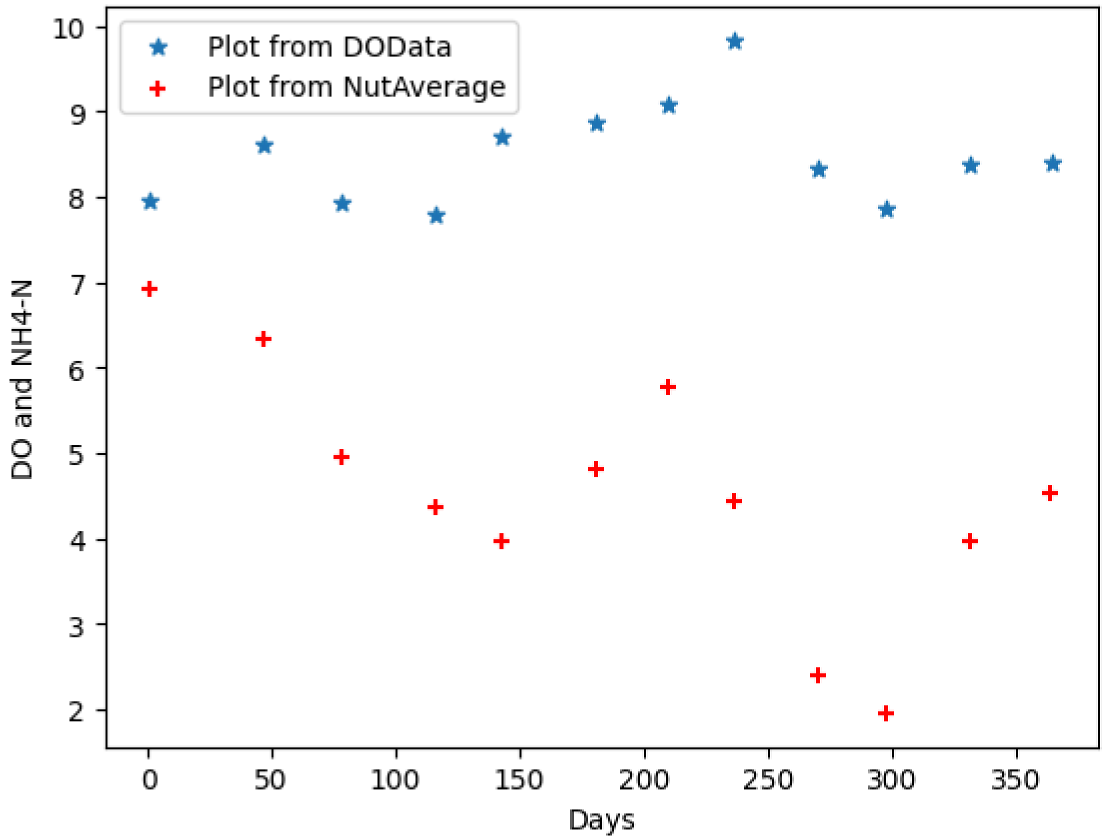
Plot the monthly DO vs NH4 (from NutAverage.xlsx) as a scatter plot.

In [23]:

```
fig, axs = plt.subplots()
axs.scatter(df['Days'],df['DO'], label="Plot from DOData", marker='*')
axs.scatter(df2['Day Count'],df2['NH4-N'], color="red", label="Plot from NutAver
axs.set_ylabel('DO and NH4-N')
axs.set_xlabel('Days')
axs.legend()
```

Out[23]:

```
<matplotlib.legend.Legend at 0x2006705f020>
```



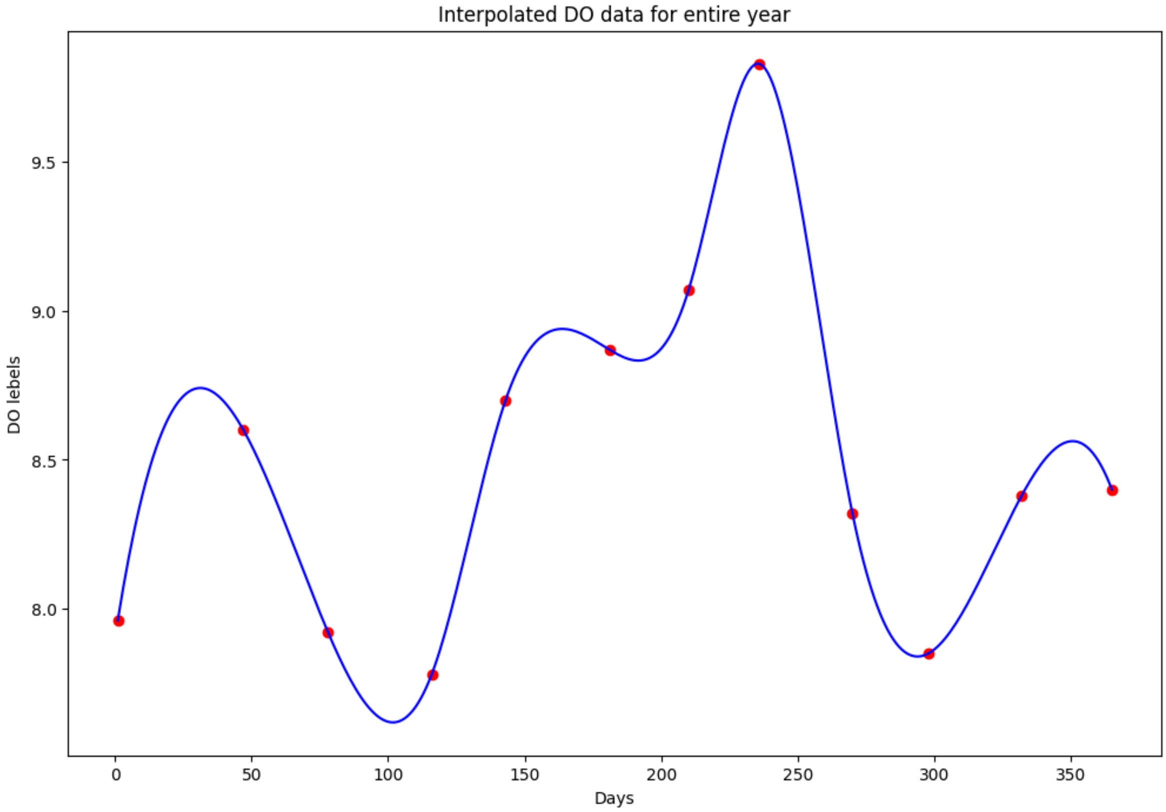
Construct an interpolating polynomial to estimate the DO for the entire year starting from day 1 to day 365. Visualise the interpolated and monthly data (with monthly data plotted as points).

In [32]:

```
from scipy.interpolate import interp1d
```

In [38]:

```
ip = interp1d(df['Days'], df['DO'], kind='cubic')
fig, axs = plt.subplots(figsize=(12,8))
days = np.arange(1,366)
axs.plot(df['Days'], df['DO'], 'or', label='Monthly Data')
axs.plot(days, ip(days), '-b', label='Interpolated Data')
axs.set_title('Interpolated DO data for entire year')
axs.set_xlabel('Days')
axs.set_ylabel('DO lebels')
plt.show()
```


Interpolated DO data for entire year
DO lebels
9.5
9.0
8.5
8.0
0
50
100
150
200
250
300
350
Days

```
In [ ]:
```