In [1]:
```
# Olivia Criscione
```

In [2]:
```
import sys
!{sys.executable} -m pip install pandas
!{sys.executable} -m pip install numpy
!{sys.executable} -m pip install matplotlib
!{sys.executable} -m pip freeze > requirements.txt
```

```
Requirement already satisfied: pandas in c:\users\olivi\downloads\cinf308fp\.venv\li
b\site-packages (2.2.2)
Requirement already satisfied: numpy>=1.23.2 in c:\users\olivi\downloads\cinf308fp\.
venv\lib\site-packages (from pandas) (2.0.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\olivi\downloads\ci
nf308fp\.venv\lib\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\olivi\downloads\cinf308fp\.v
env\lib\site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\olivi\downloads\cinf308fp
\.venv\lib\site-packages (from pandas) (2024.1)
Requirement already satisfied: six>=1.5 in c:\users\olivi\downloads\cinf308fp\.venv
\lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Requirement already satisfied: numpy in c:\users\olivi\downloads\cinf308fp\.venv\lib
\site-packages (2.0.1)
Requirement already satisfied: matplotlib in c:\users\olivi\downloads\cinf308fp\.ven
v\lib\site-packages (3.9.1.post1)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\olivi\downloads\cinf308f
p\.venv\lib\site-packages (from matplotlib) (1.2.1)
Requirement already satisfied: cycler>=0.10 in c:\users\olivi\downloads\cinf308fp\.v
env\lib\site-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\olivi\downloads\cinf308
fp\.venv\lib\site-packages (from matplotlib) (4.53.1)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\olivi\downloads\cinf308
fp\.venv\lib\site-packages (from matplotlib) (1.4.5)
Requirement already satisfied: numpy>=1.23 in c:\users\olivi\downloads\cinf308fp\.ve
nv\lib\site-packages (from matplotlib) (2.0.1)
Requirement already satisfied: packaging>=20.0 in c:\users\olivi\downloads\cinf308fp
\.venv\lib\site-packages (from matplotlib) (24.1)
Requirement already satisfied: pillow>=8 in c:\users\olivi\downloads\cinf308fp\.venv
\lib\site-packages (from matplotlib) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\olivi\downloads\cinf308f
p\.venv\lib\site-packages (from matplotlib) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\olivi\downloads\cinf
308fp\.venv\lib\site-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in c:\users\olivi\downloads\cinf308fp\.venv
\lib\site-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
```

In [3]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

In this project, we are importing the following libraries: `pandas` for data manipulation and analysis. `matplotlib.pyplot` for creating static, animated, and interactive visualizations. `seaborn` for making statistical graphics based on matplotlib. 'numpy' for ...

In [4]:
```
df = pd.read_csv("Excelsior_Scholarship_Recipients_and_Dollars_by_College_Code_2017
df.head()
```

Out[4]:

	Academic Year	Scholarship Name	Sector Type	TAP Sector Group	TAP College Code	Federal School Code	TAP College Name	Scholarship Headcount	Sch
0	2021	Excelsior Scholarship Program	PUBLIC	1-CUNY SR	1409	7273	CUNY BARUCH COLLEGE	1004	2
1	2021	Excelsior Scholarship Program	PUBLIC	1-CUNY SR	1410	2687	CUNY BROOKLYN COLLEGE	592	2
2	2021	Excelsior Scholarship Program	PUBLIC	1-CUNY SR	1411	2688	CUNY CITY COLLEGE	637	2
3	2021	Excelsior Scholarship Program	PUBLIC	1-CUNY SR	1417	2698	CUNY COL STATEN ISLAND	375	1
4	2021	Excelsior Scholarship Program	PUBLIC	1-CUNY SR	1420	4765	CUNY GRAD SCH UNDERGRAD PROG	5	

The dataset "Excelsior_Scholarship_Recipients_and_Dollars_by_College_Code_2017_2021.csv" has been imported using pandas' `read_csv` function.

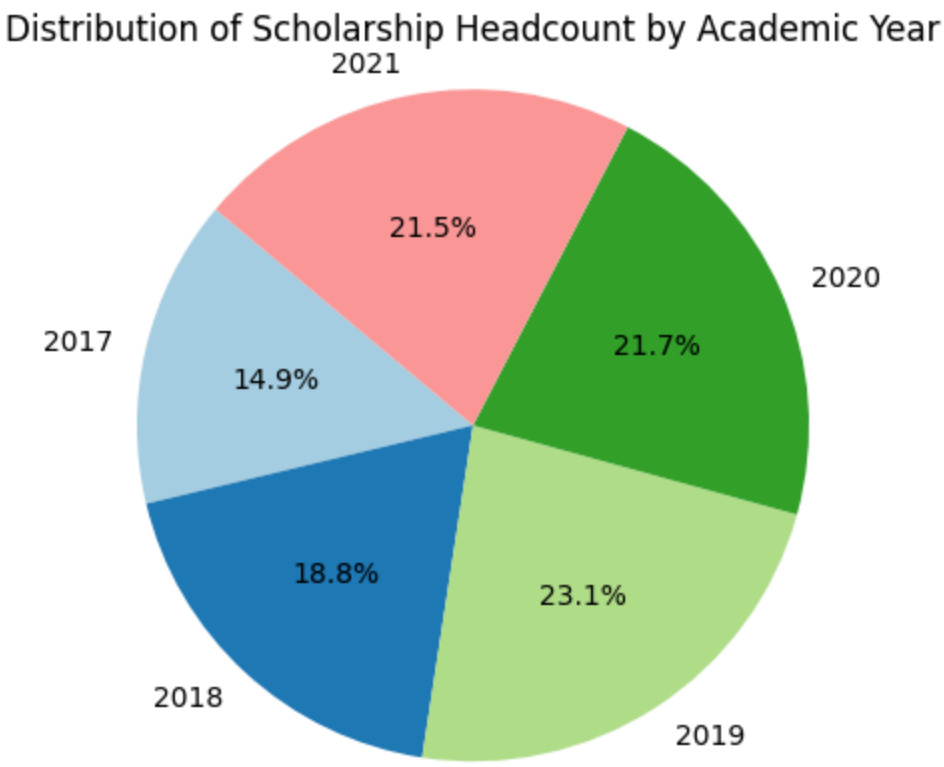
In [5]:
```
# Example data cleaning
df_cleaned = df.dropna()  # Removing any rows with missing values
df_cleaned.head()
```

Out[5]:

	Academic Year	Scholarship Name	Sector Type	TAP Sector Group	TAP College Code	Federal School Code	TAP College Name	Scholarship Headcount	Sch
0	2021	Excelsior Scholarship Program	PUBLIC	1-CUNY SR	1409	7273	CUNY BARUCH COLLEGE	1004	2
1	2021	Excelsior Scholarship Program	PUBLIC	1-CUNY SR	1410	2687	CUNY BROOKLYN COLLEGE	592	2
2	2021	Excelsior Scholarship Program	PUBLIC	1-CUNY SR	1411	2688	CUNY CITY COLLEGE	637	2
3	2021	Excelsior Scholarship Program	PUBLIC	1-CUNY SR	1417	2698	CUNY COL STATEN ISLAND	375	1
4	2021	Excelsior Scholarship Program	PUBLIC	1-CUNY SR	1420	4765	CUNY GRAD SCH UNDERGRAD PROG	5	

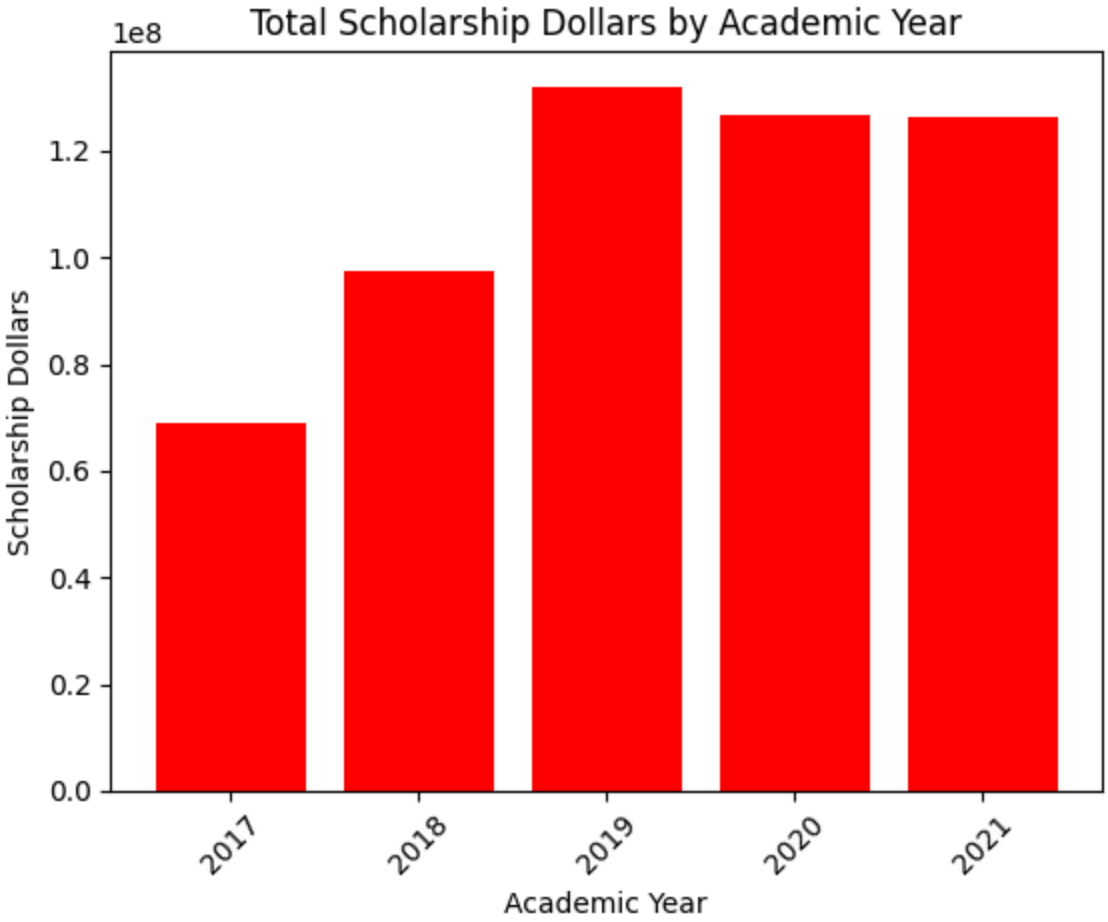
In the above cell I cleaned the data

In [6]:
```
# Groups by 'Academic Year' and sum 'Scholarship Headcount'
pie_data = df_cleaned.groupby('Academic Year')['Scholarship Headcount'].sum()

# Checks the prepared data
plt.pie(pie_data, labels=pie_data.index, autopct='%1.1f%%', startangle=140, colors=
plt.title('Distribution of Scholarship Headcount by Academic Year')
plt.axis('equal')  # ensures the pie chart is circular.
plt.show()
```



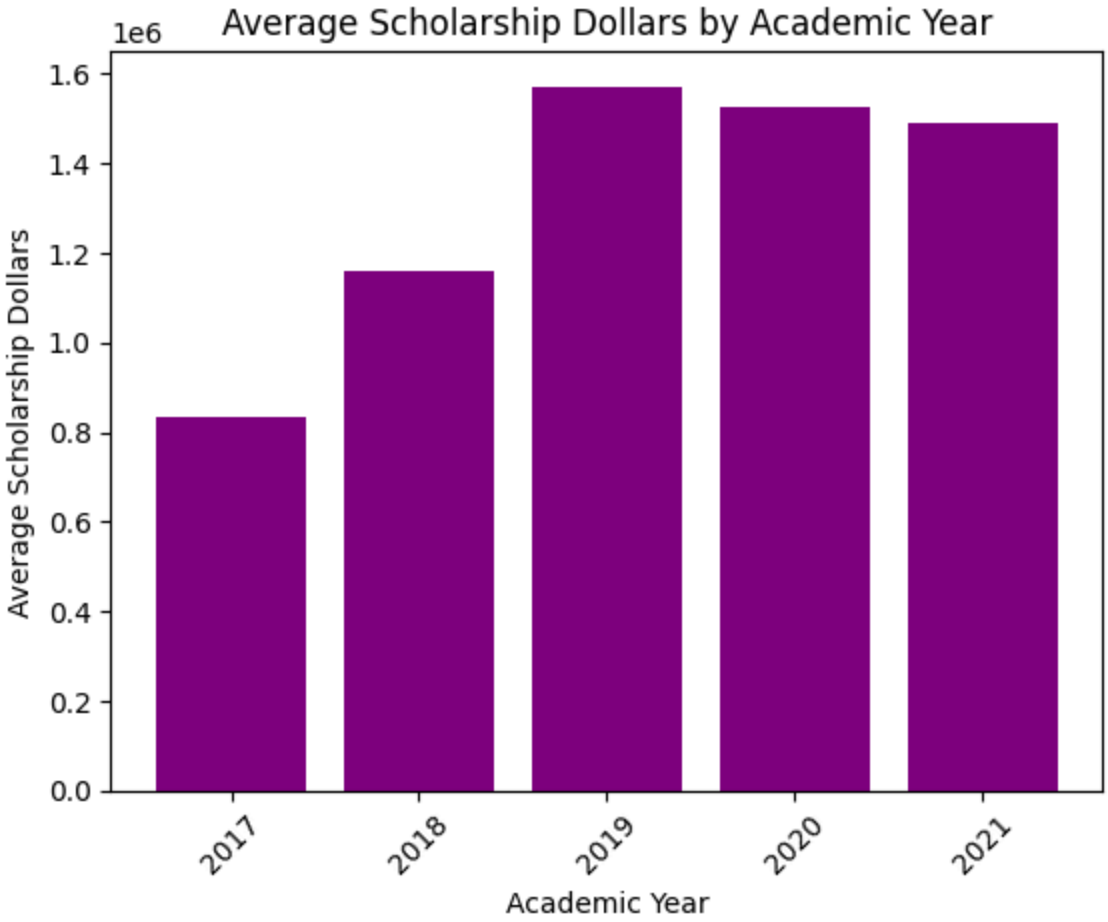
This plot above shows how the number of scholarship recipients changes over different academic years using a piechart. A piechart is ideal for showing how scholarship headcount changes over academic years because it clearly illustrates trends and patterns over time by providing the perentages of each year.

In [7]:
```
# Group by 'Academic Year' and sum 'Scholarship Dollars'
bar_data = df_cleaned.groupby('Academic Year')['Scholarship Dollars'].sum()

# Create the bar plot
plt.bar(bar_data.index, bar_data.values, color='red')  # Bar plot
plt.title('Total Scholarship Dollars by Academic Year')
plt.xlabel('Academic Year')
plt.ylabel('Scholarship Dollars')
plt.xticks(rotation=45)  # for better readability
plt.show()
```



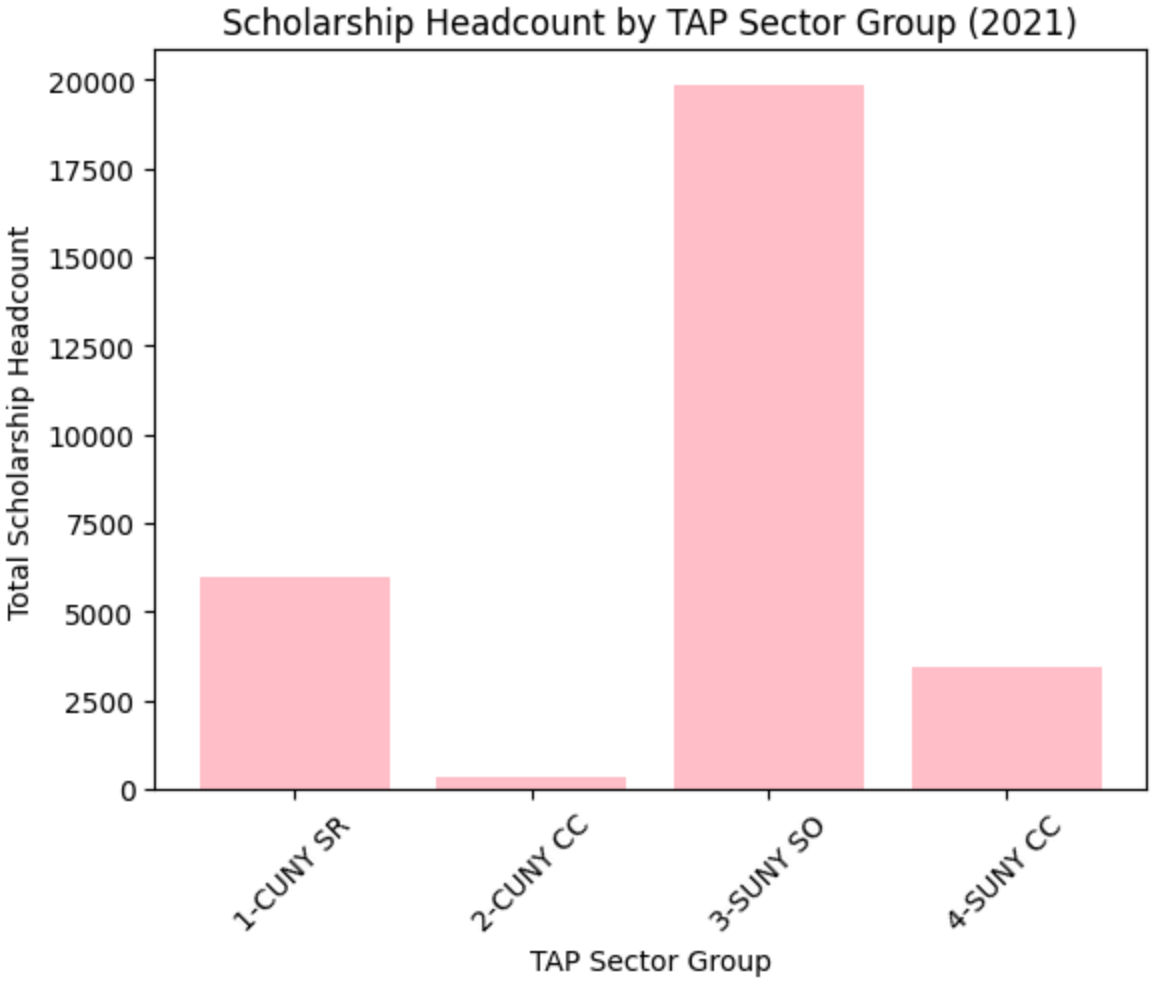
This plot shows the total scholarship dollars distributed from 2017 - 2021. I decided to use a bar plot for this graph because it best shows the relationship of scholarship dollars between different academic years. This type of graph clearly shows the data for each year including the fact that 2019 had the most total scholarship dollars.

In [8]:
```
# Computes the average scholarship dollars by academic year
average_scholarship = df_cleaned.groupby('Academic Year')['Scholarship Dollars'].me

# Creates the bar plot for the average scholarship dollars by year
plt.bar(average_scholarship.index, average_scholarship.values, color='purple')
plt.title('Average Scholarship Dollars by Academic Year')
plt.xlabel('Academic Year')
plt.ylabel('Average Scholarship Dollars')
plt.xticks(rotation=45)  # for readability
plt.show()
```



This plot shows the average scholarship dollars awarded each academic year using a bar plot. Just like the previous example, I decided to use a bar plot because it represents the data the best. It is really easy to visually interpret the data and the fact that the 2019 academic year had the highest average scholarship dollars.

In [9]:
```
# Filter for 2021 year
df_filtered = df_cleaned[df_cleaned['Academic Year'] == 2021]

# Computes the total scholarship headcount by TAP Sector Group for 2021
total_headcount = df_filtered.groupby('TAP Sector Group')['Scholarship Headcount'].

# Creates the bar plot
plt.bar(total_headcount.index, total_headcount.values, color='pink')
plt.title('Scholarship Headcount by TAP Sector Group (2021)')
plt.xlabel('TAP Sector Group')
plt.ylabel('Total Scholarship Headcount')
plt.xticks(rotation=45)  # for better readability
plt.show()
```

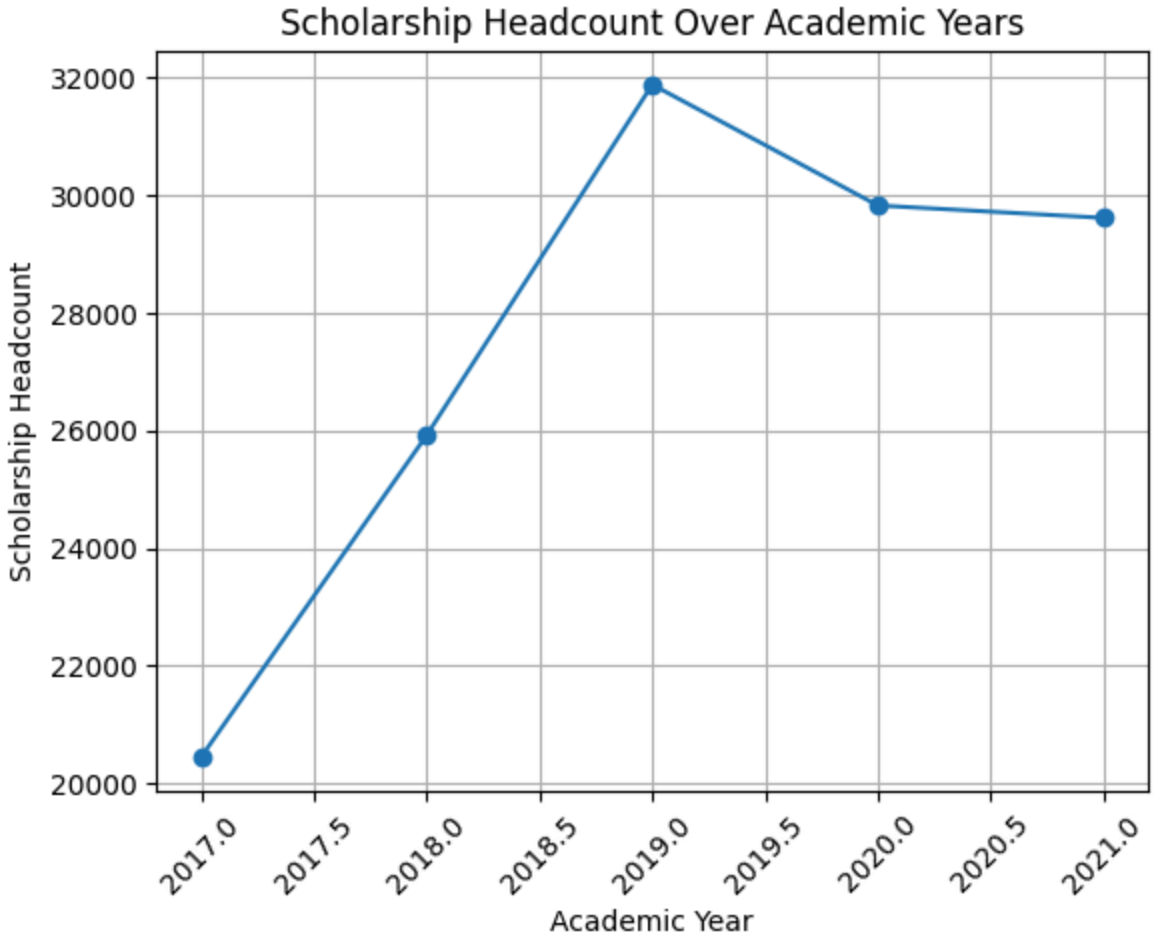


This plot shows the number of scholarship recipients in each TAP Sector Group for the 2021 academic year using a bar plot. I think that the bar plot does a great job of showing the relationship between each tap sector and the scholarship headcount for 2021. I think the only other good plot type that could represent this data well would be a piechart.

In [10]:
```
# Creating the scatter plot
plt.scatter(df_cleaned['Scholarship Headcount'], df_cleaned['Scholarship Dollars'],
plt.title('Relationship Between Scholarship Headcount and Scholarship Dollars')
plt.xlabel('Scholarship Headcount')
plt.ylabel('Scholarship Dollars')
plt.grid(True, linestyle='--', alpha=0.7)  # Adds grid lines to make the graph easi
plt.show()
```

This scatter plot shows the relationship between the number of scholarship recipients and the total scholarship dollars. I think that a scatter plot best shows this data set because it clearly shows the outliers as well as the most common scholarship dollars. I also like how it is clear how common the data points near the origin are because they are shown as a darker organe since there are so many data points there.

In [11]:
```
headcount_per_year = df_cleaned.groupby('Academic Year')['Scholarship Headcount'].s

plt.plot(headcount_per_year['Academic Year'], headcount_per_year['Scholarship Headc
plt.title('Scholarship Headcount Over Academic Years')
plt.xlabel('Academic Year')
plt.ylabel('Scholarship Headcount')
plt.grid(True)
plt.xticks(rotation=45)
plt.show()
```



The line plot above shows the relationship of scholarship headcount over academic years. The line plot is the best type of plot to use for showing this data because it show the trend in scholarship headcount from 2017 to 2021. The relationship of this graph increases from 2017 to 2019 and then decreases from 2019 to 2021.