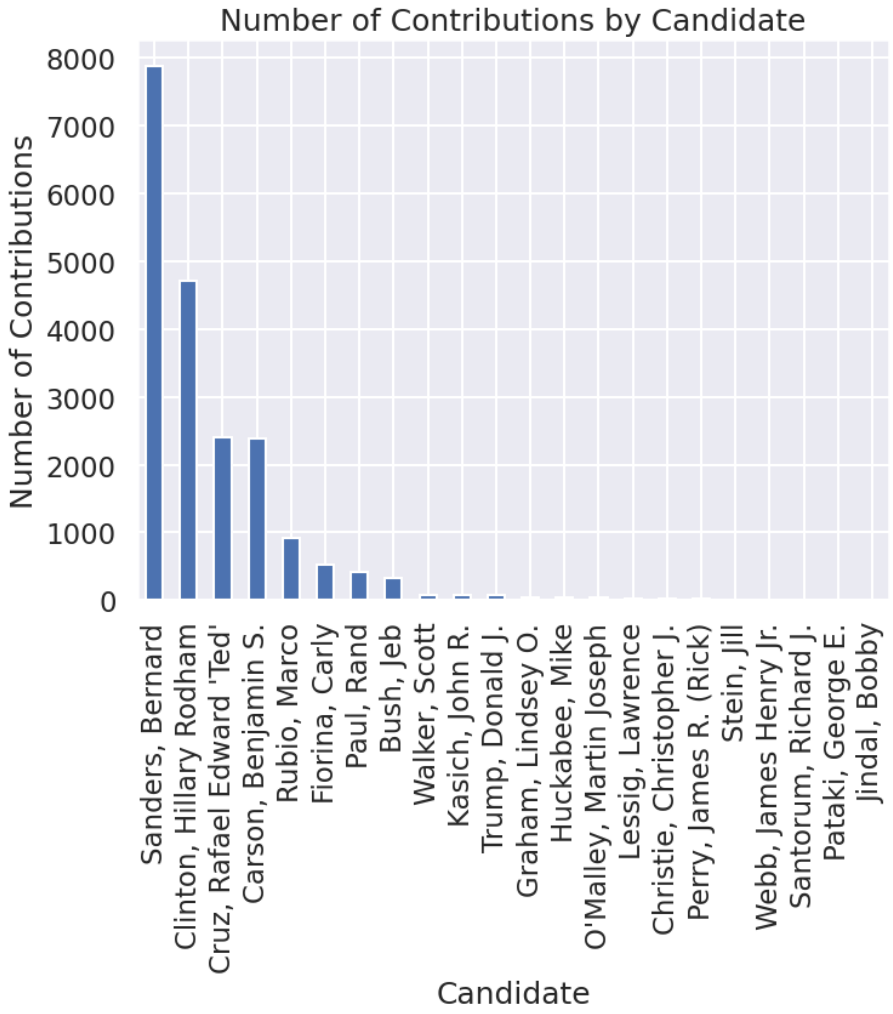
```
#@ 1 Show the number of contributions by candidate using a barplot.
# Hint: use value_counts(), and use a pandas bar plot.
contributions_by_candidate = df['cand_nm'].value_counts()
contributions_by_candidate.plot(kind='bar')
plt.title('Number of Contributions by Candidate')
plt.xlabel('Candidate')
plt.ylabel('Number of Contributions')
```

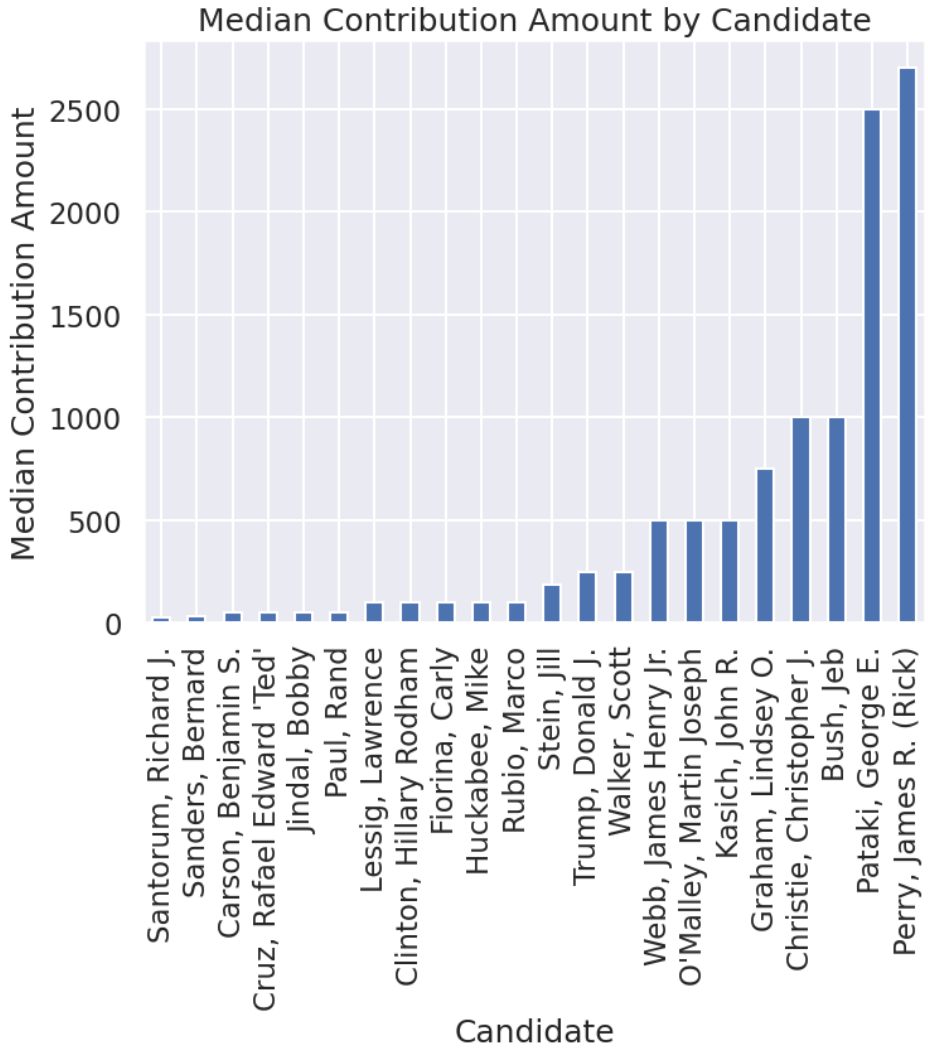
```
Text(0, 0.5, 'Number of Contributions')
```



Let's look at the amount of the contributions, instead of the number of contributions. Which candidates had the highest median contribution amounts?

```
#@ 2 Show the median contribution amount by candidate.
median_cont_by_cand = df.groupby('cand_nm')['contb_receipt_amt'].median()
median_cont_by_cand = median_cont_by_cand.sort_values()
median_cont_by_cand.plot(kind='bar')
plt.title('Median Contribution Amount by Candidate')
plt.xlabel('Candidate')
plt.ylabel('Median Contribution Amount')
```

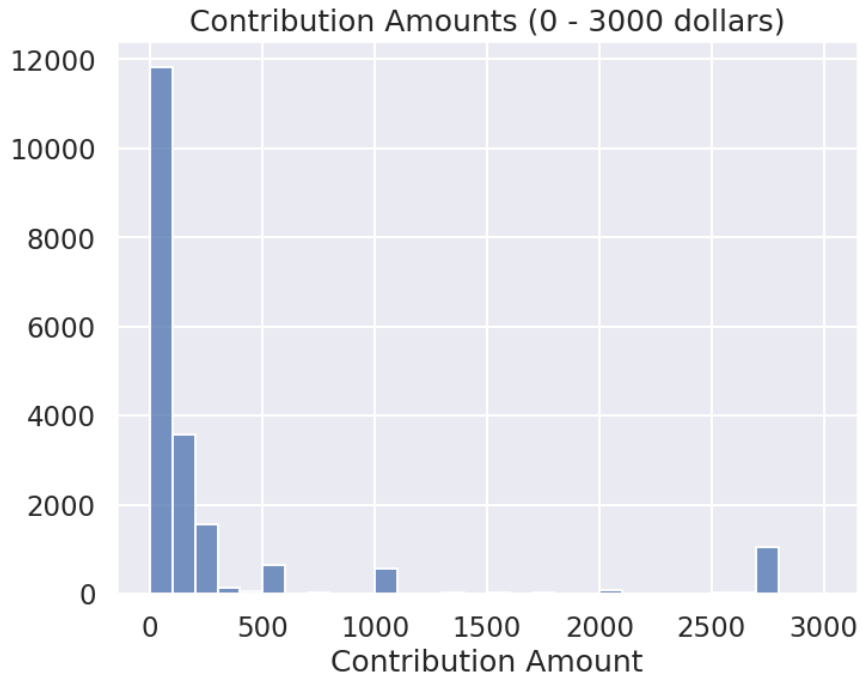
```
Text(0, 0.5, 'Median Contribution Amount')
```



What is the distribution of the contribution amounts? There are a small number of very large amounts, which make it hard to display the distribution. Also, there are some negative contribution amounts that seem to reflect returned contributions. Therefore, let's focus on contributions ranging from 0 to 3,000 dollars.

```
#@ 3 Create a histogram showing contribution amounts.  Show
# contributions from 0 - 3000 dollars only.  Create the
# histogram with Seaborn.
cont_3k = df[(df['contb_receipt_amt'] >= 0) & (df['contb_receipt_amt'] <= 3000)]

sns.histplot(cont_3k['contb_receipt_amt'], bins=30, kde=False)
plt.title('Contribution Amounts (0 - 3000 dollars)')
plt.xlabel('Contribution Amount')
plt.ylabel('')
```

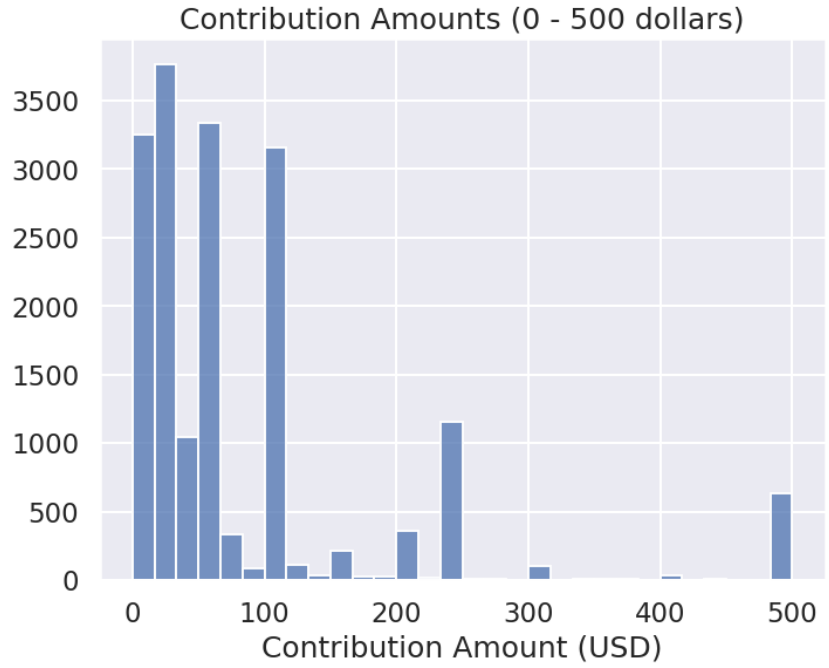
```
Text(0, 0.5, '')
```



It appears that most contributions are small. Let's restrict our attention to an even smaller range of contributions to get a better idea of how small contributions are distributed.

```
[99] #@ 4 Create a histogram showing contribution amounts.  Show
     # contributions from 0 - 500 dollars only.  Create the
     # histogram with Seaborn.
     cont_500 = df[(df['contb_receipt_amt'] >= 0) & (df['contb_receipt_amt'] <= 500)]

     sns.histplot(cont_500['contb_receipt_amt'], bins=30, kde=False)
     plt.title('Contribution Amounts (0 - 500 dollars)')
     plt.xlabel('Contribution Amount (USD)')
     plt.ylabel('')
```

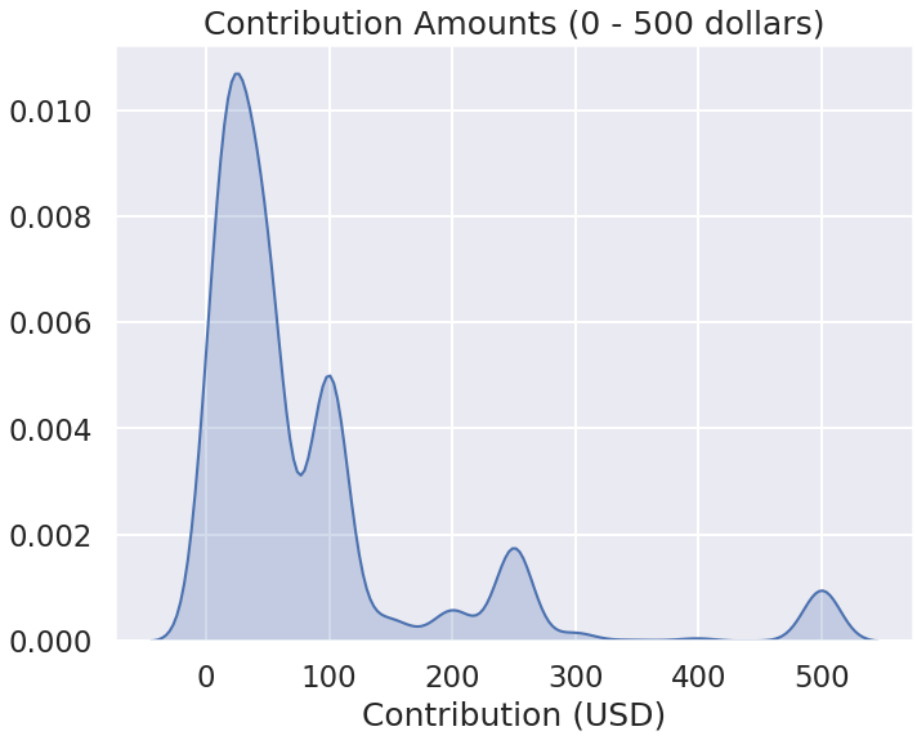
```
Text(0, 0.5, '')
```



The appearance of a histogram is sensitive to the number of bins that are used and where the bin edges lie. Let's look at the contribution amounts again using a density plot.

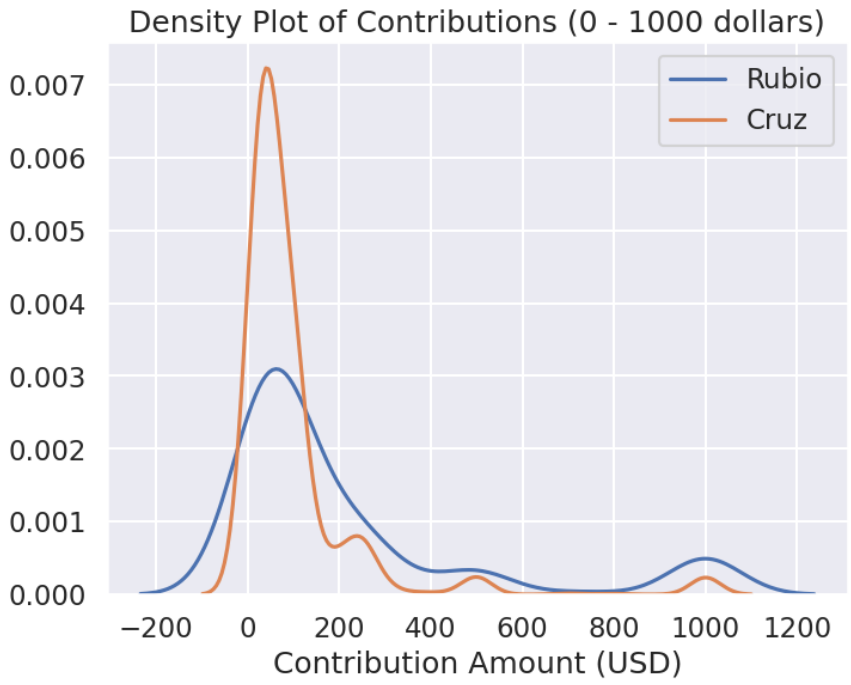
```python
#@ 5 Create a density plot (sometimes called a kernel density
# plot) showing contribution amounts.  Show contributions from
# 0 - 500 dollars only.  Create the density plot with Seaborn.
# histogram use Seaborn.
# Hint: you may want to start by creating a series containing
# the contb_receipt_amt values from 0-500.
cont_500 = df[(df['contb_receipt_amt'] >= 0) & (df['contb_receipt_amt'] <= 500)]
cont_500 = cont_500['contb_receipt_amt']

sns.kdeplot(cont_500, fill=True)
plt.title('Contribution Amounts (0 - 500 dollars)')
plt.xlabel('Contribution (USD)')
plt.ylabel('')
```

```
Text(0, 0.5, '')
```



Let's compare the size of contributions between candidates Rubio and Cruz. Did one of them tend to get larger-sized contributions?

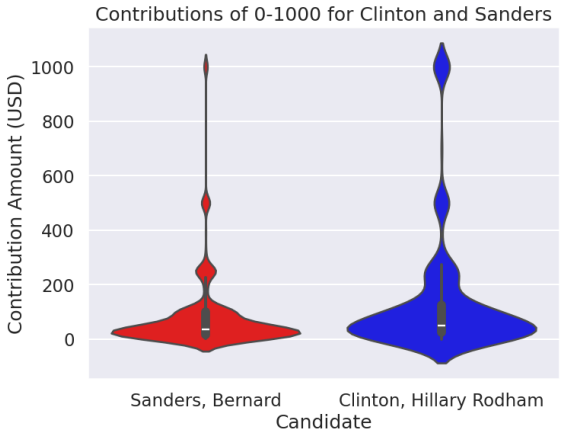
```python
from itertools import filterfalse
#@ 6 Create a "double density plot" showing the contributions for
# Rubio and Cruz.  Show contributions in the range of 0-1000 dollars
# only.  Be sure to include a legend.
# Hint: you can create two series, one for 0-1000 contributions to
# Rubio, and another for 0-1000 contributions to Cruz.
# Remember that you can superimpose plots by simply plotting one
# after another.
rubio = df[(df['cand_nm'] == 'Rubio, Marco') & (df['contb_receipt_amt'] >= 0) & (df['contb_receipt_amt'] <= 1000)]
cruz = df[(df['cand_nm'] == "Cruz, Rafael Edward 'Ted'") & (df['contb_receipt_amt'] >= 0) & (df['contb_receipt_amt'] <= 1000)]

rubio_series = rubio['contb_receipt_amt']
cruz_series = cruz['contb_receipt_amt']

sns.kdeplot(rubio_series, fill=False, label='Rubio')
sns.kdeplot(cruz_series, fill=False, label='Cruz')
plt.title('Density Plot of Contributions (0 - 1000 dollars)')
plt.xlabel('Contribution Amount (USD)')
plt.ylabel('')
plt.legend()
plt.show()
```



Rubio and Cruz were Republican candidates. Let's look at a pair of Democratic candidates.

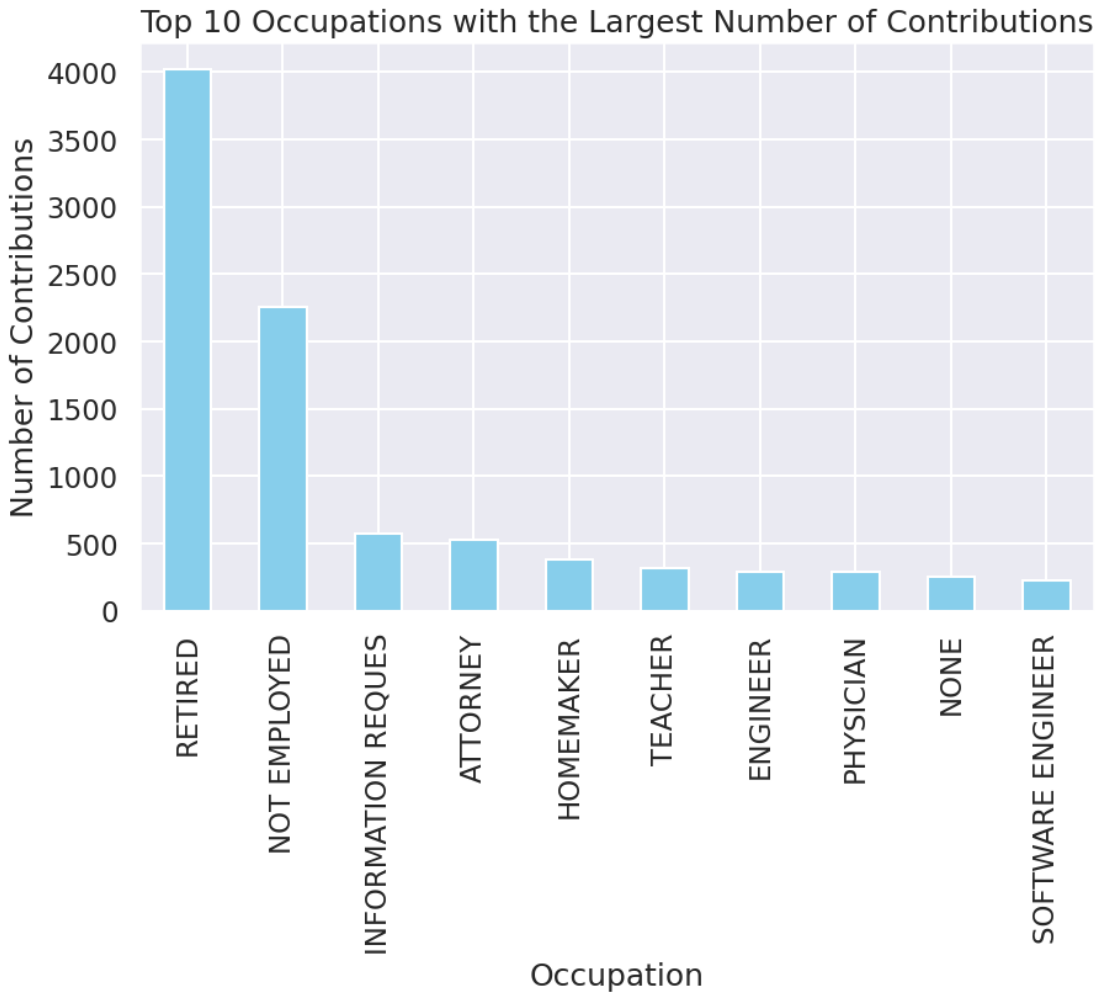
```
#@ 7 Show the contributions of 0-1000 for Clinton and Sanders.
# Use a seaborn violin plot.
# Hint: create a modified version of the data frame that contains only
# contributions for Sanders and Clinton, and only contains contributions
# from 0 to 1000 dollars.  Then use Seaborn's violinplot.
clinton_sanders_contributions = df[((df['cand_nm'] == 'Clinton, Hillary Rodham') | (df['cand_nm'] == 'Sanders, Bernard')) & (df['contb_receipt_amt'] >= 0) & (df['contb_receipt_amt'] <= 1000)]
sns.violinplot(x='cand_nm', y='contb_receipt_amt', hue='cand_nm', data=clinton_sanders_contributions, palette={'Clinton, Hillary Rodham': 'blue', 'Sanders, Bernard': 'red'})
plt.title('Contributions of 0-1000 for Clinton and Sanders')
plt.xlabel('Candidate')
plt.ylabel('Contribution Amount (USD)')
```

```
Text(0, 0.5, 'Contribution Amount (USD)')
```



```
[95] #@ 8 Create a bar plot showing th total number of contributions by occupation,
     # for the 10 occupations with the largest number of contributions.  Use
     # Pandas for the bar plot.  Limit the occupation names to 18 characters.
     # Hint: to limit the occupation names to 18 characters, you can create a
     # new column 'short_occ' by using pd.Series.str.slice on the
     # 'contbr_occupation' column.
     df['short_occ'] = df['contbr_occupation'].str.slice(0, 18)
     occupation_counts = df['short_occ'].value_counts()
     top_10 = occupation_counts.head(10)
     top_10.plot(kind='bar', figsize=(10, 6), color='skyblue')
     plt.title('Top 10 Occupations with the Largest Number of Contributions')
     plt.xlabel('Occupation')
     plt.ylabel('Number of Contributions')
```

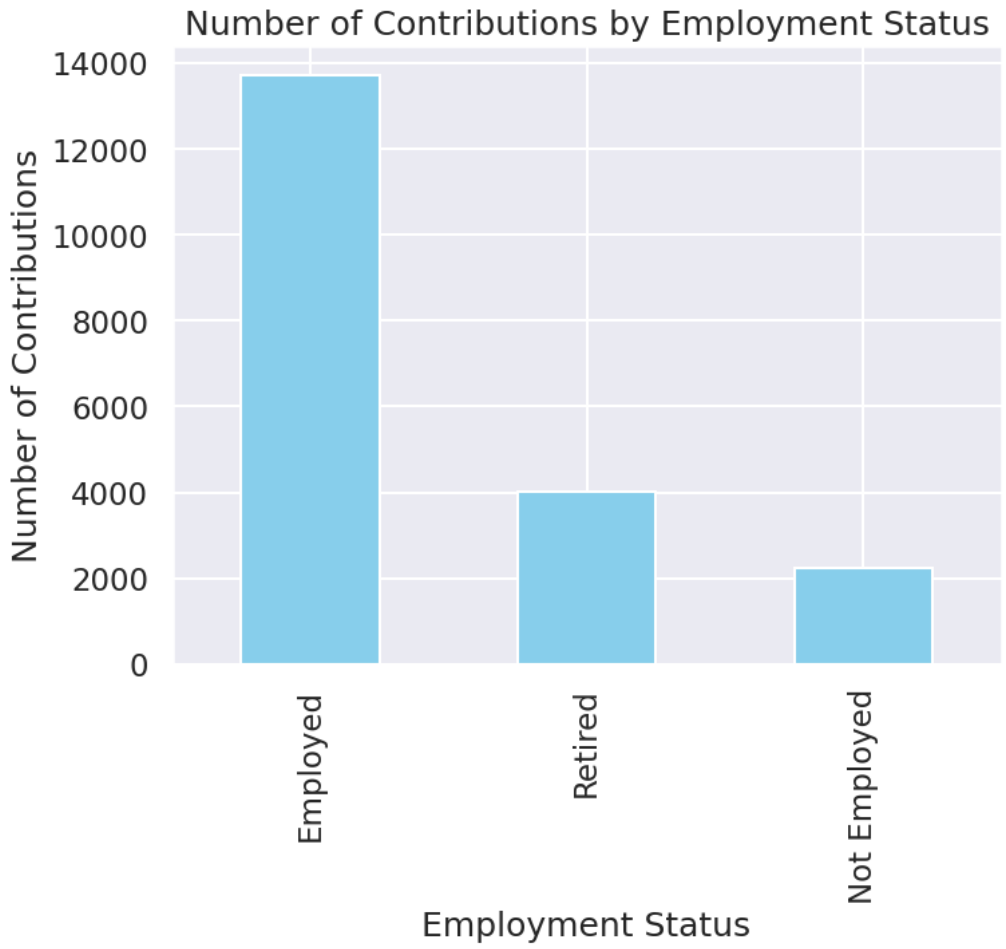
```
Text(0, 0.5, 'Number of Contributions')
```



We can classify contributors as either employed, unemployed, or retired. Among these groups, which makes the most contributions?

```
#@ 9 Create a new column "employment_status", derived from the
# contbr_occupation column.  The value of employment_status should
# be "EMPLOYED" if contbr_occupation is not "RETIRED" or "NOT EMPLOYED",
# and should be the original contbr_occupation otherwise.  Show the
# number of contributions by employment status as a bar plot.
# Hint: to create the new column, consider creating a function that
# takes as input a contbr_occupation value and returns an employement
# status value.  Then use this function with 'apply'.
def employment_status(occupation):
    if occupation == 'RETIRED':
        return 'Retired'
    elif occupation == 'NOT EMPLOYED':
        return 'Not Employed'
    else:
        return 'Employed'

df['employment_status'] = df['contbr_occupation'].apply(employment_status)

# Group the data by "employment_status" and count the number of contributions for each employment status
employment_status = df['employment_status'].value_counts()

employment_status.plot(kind='bar', figsize=(8,6), color='skyblue')
plt.title('Number of Contributions by Employment Status')
plt.xlabel('Employment Status')
plt.ylabel('Number of Contributions')
```

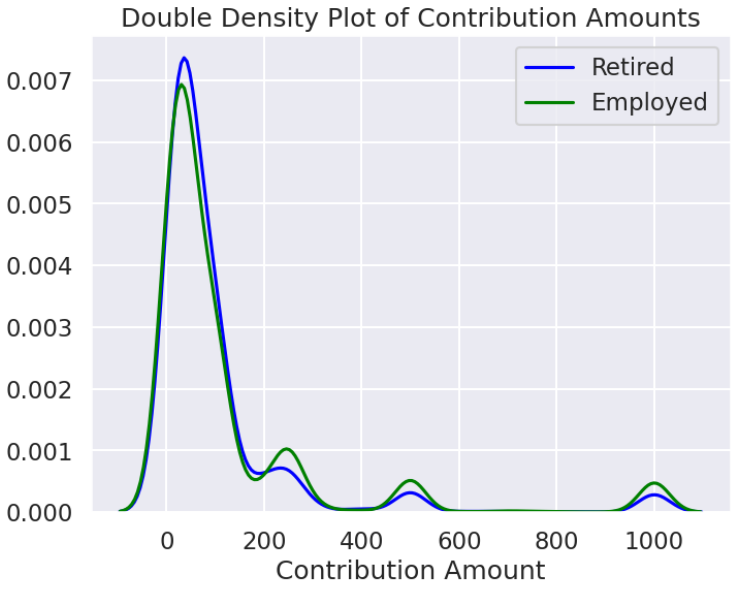
```
Text(0, 0.5, 'Number of Contributions')
```



Do retired contributors tend to make smaller contributions than employed contributors? It seems likely, but what does the data say?

```
#@ 10 Create a double density plot showing the distribution of
# contribution amounts from those with employment_status values
# of RETIRED and EMPLOYED.  Include only contributions of $0-1000.
# Use Seaborn, and make sure to include
# a legend.
# Hint: consider creating two series, one for the contributions
# from retired contributors, and one for the contributions from
# employed contributors.
contributions_emp = df[(df['employment_status'].isin(['Retired', 'Employed'])) & (df['contb_receipt_amt'] >= 0) & (df['contb_receipt_amt'] <= 1000)]

#print(contributions_emp.head())
#print(contributions_emp['employment_status'].unique())
#print("Number of rows in filtered dataset:", len(contributions_emp))

retired_contributions = contributions_emp[contributions_emp['employment_status'] == 'Retired']['contb_receipt_amt']
employed_contributions = contributions_emp[contributions_emp['employment_status'] == 'Employed']['contb_receipt_amt']

sns.kdeplot(retired_contributions, fill=False, label='Retired', color='blue')
sns.kdeplot(employed_contributions, fill=False, label='Employed', color='green')
plt.title('Double Density Plot of Contribution Amounts')
plt.xlabel('Contribution Amount')
plt.ylabel('')
plt.legend()
```

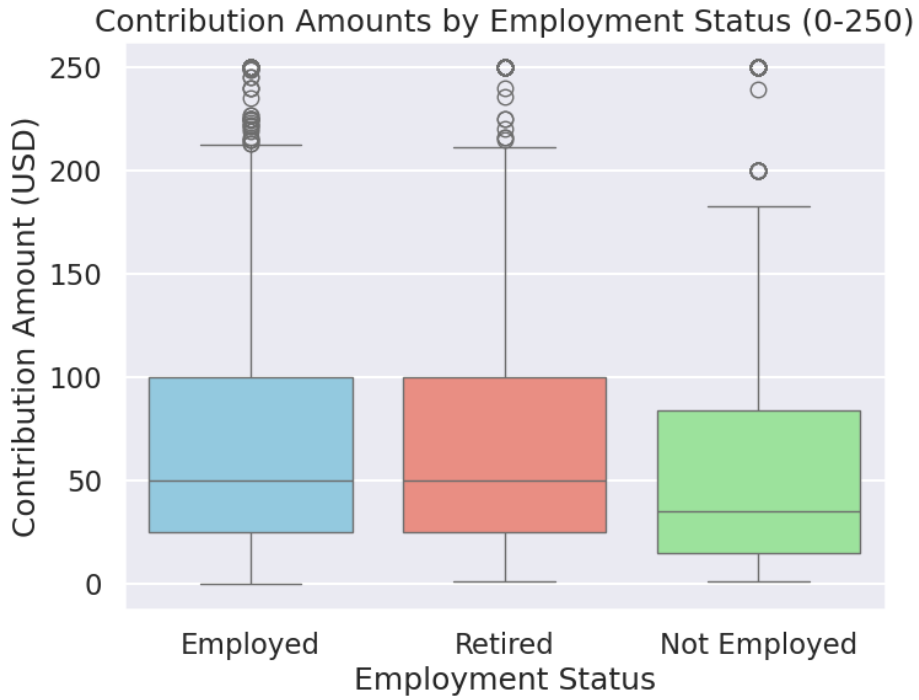
```
<matplotlib.legend.Legend at 0x7fa88a5f6380>
```



It appears that contributions from the retired and the employed are pretty similar, although there is a significant difference when you focus on larger contributions. Let's look more into the size of contributions from those who are employed, retired, or unemployed.

```
#@ 11 Create a box plot of contribution amounts for each employment
# status category.  Use Seaborn to create the bar plot, and show
# only contributions in the range of $0-250.
# Hint: you may want to create a version of df that contains only
# contributions in the 0-250 range.

df_range = df[(df['contb_receipt_amt'] >= 0) & (df['contb_receipt_amt'] <= 250)]

colors = {"Employed": "skyblue", "Retired": "salmon", "Not Employed": "lightgreen"}

sns.boxplot(x='employment_status', y='contb_receipt_amt', hue='employment_status', data=df_range, palette=colors)
plt.title('Contribution Amounts by Employment Status (0-250)')
plt.xlabel('Employment Status')
plt.ylabel('Contribution Amount (USD)')
```

```
Text(0, 0.5, 'Contribution Amount (USD)')
```



Previously we looked at the number of contributions from different occupations. What about the size of contributions from different occupations? Let's focus on a few occupations that contribute a lot.

```
#@ 12 Create a bar plot showing the average contribution amount
# for the occupations 'ATTORNEY', 'TEACHER', 'ENGINEER' and 'PHYSICIAN'.
# Include contributions of any amount.  Use Pandas to create the bar plot.
# Show the occupations in decreasing order of mean contribution amount.
# Hint: you may want to create a new data frame that is like df except
# that it only includes data associated with the four occupations.
# To do this, consider the Pandas method pd.Series.isin

occupations = df[df['contbr_occupation'].isin(['ATTORNEY', 'TEACHER', 'ENGINEER', 'PHYSICIAN'])]

mean_contributions = occupations.groupby('contbr_occupation')['contb_receipt_amt'].mean()

contributions_sorted = mean_contributions.sort_values(ascending=False)

contributions_sorted.plot(kind='bar', color='skyblue')
plt.title('Average Contribution Amount by Occupation')
plt.xlabel('Occupation')
plt.ylabel('Avg Contribution Amount (USD)')
```

```
Text(0, 0.5, 'Avg Contribution Amount (USD)')
```