Hard Level Task 2

Present your findings on the final project, where you are tasked with creating a Jupyter notebook from scratch and conducting a data analysis on a dataset of your choice. This comprehensive process involves selecting a dataset that piques your interest, exploring its contents within a Jupyter notebook, and identifying research questions that the data might help answer. Guidelines:

1. Begin by finding a dataset that piques your interest. You can choose from a list of places with valuable datasets provided in our reading, or feel free to select data related to your hobbies or work if it is publicly available.
2. Explore the dataset in a Jupyter notebook, gaining a deep understanding of its contents. This exploration phase will help you identify the types of questions that can be addressed using the available data. Rememb

```
import pandas as pd
import matplotlib.pyplot as plt
!pip install num2words
from num2words import num2words
```

```
Collecting num2words
  Downloading num2words-0.5.13-py3-none-any.whl (143 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 143.3/143.3 kB 3.6 MB/s eta 0:00:00
Collecting docopt>=0.6.2 (from num2words)
  Downloading docopt-0.6.2.tar.gz (25 kB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: docopt
  Building wheel for docopt (setup.py) ... done
  Created wheel for docopt: filename=docopt-0.6.2-py2.py3-none-any.whl size=13706 sha256=737fabeae8f99e6679782613618d3a55ba86f2ad450371202dab204db61af9f2
  Stored in directory: /root/.cache/pip/wheels/fc/ab/d4/5da2067ac95b36618c629a5f93f809425700506f72c9732fac
Successfully built docopt
Installing collected packages: docopt, num2words
Successfully installed docopt-0.6.2 num2words-0.5.13
```

```

donated = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ShadowFox Internship/datasets/donor_data.csv')
received = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ShadowFox Internship/datasets/rec_data.csv')
```

Double-click (or enter) to edit

Knowing Data

- Total number of columns
- Total number of rows
- Structure of dataset
- Datatypes of each column
- Donater List (Company list who donated)
- Receiver List (Political party list who encashed)

```
donated_columnList = donated.columns
donated_columnCount = len(donated.columns)

print("Donated Column: ", donated_columnList);
print("Total Count: ", donated_columnCount)
```

```
Donated Column:  Index(['SNo', 'Urn', 'JournalDate', 'PurchaseDate', 'ExpiryDate', 'Purchaser',
       'Prefix', 'BondNumber', 'Denominations', 'PayBranchCode', 'PayTeller'],
      dtype='object')
Total Count:  11
```

```
donated_rowsCount = len(donated)
print("Total Rows Count: ", donated_rowsCount)
```

```
Total Rows Count:  18871
```

```
received_rowsCount = len(received)
print("Total Rows Count: ", received_rowsCount)
```

```
Total Rows Count:  20421
```

```
received_columnList = received.columns
received_columnCount = len(donated.columns)

print("Donated Column: ", received_columnList);
print("Total Count: ", received_columnCount)
```

```
Donated Column:  Index(['Sno', 'DateEncashment', 'PartyName', 'AccountNum', 'Prefix',
       'BondNumber', 'Denominations', 'PayBranchCode', 'PayTeller'],
      dtype='object')
Total Count:  11
```

```
donated.dtypes #Here we can observe that Date is treated as Object, we can converted into pandas datatype for better operation in Data Cleaning Part
```

```
SNo               int64
Urn              object
JournalDate      object
PurchaseDate     object
ExpiryDate       object
Purchaser        object
Prefix           object
BondNumber        int64
Denominations     int64
PayBranchCode     int64
PayTeller         int64
dtype: object
```

```
received.dtypes #Here we can observe that Date is treated as Object, we can converted into pandas datatype for better operation in Data Cleaning Part
```

```
Sno                int64
DateEncashment    object
PartyName         object
AccountNum        object
Prefix            object
BondNumber         int64
Denominations      int64
PayBranchCode      int64
```

```
PayTeller          int64
dtype: object
```

```python
donated_unique_companies = donated['Purchaser'].drop_duplicates()
print("Total Count of Companies: ", len(donated_unique_companies))
donated_unique_companies
```

```
Total Count of Companies:  1316
0                                    A B C INDIA LIMITED
13       ACROPOLIS MAINTENANCE SERVICES PRIVATE LIMITED
20                                   ARIHANT ENTERPRISES
24                                    CHOUDHARY GARMENTS
26                             ESSEL MINING AND INDS LTD
                              ...
18800          SRI CHAITANYA STUDENTS FACILITY MANAGEME
18815               SYLVANUS BUILDERS AND DEVELOPERS LI
18821                   VEDIKA VANIJYA PVT LTD-SELF A/C
18826                                       VIDUR GUPTA
18831              VIHAAN AUTO VENTURES PRIVATE LIMITED
Name: Purchaser, Length: 1316, dtype: object
```

```python
received_unique_parties = received['PartyName'].drop_duplicates()
print("Total Count of Political Parties: ", len(received_unique_parties))
received_unique_parties
```

```
Total Count of Political Parties:  24
0                 ALL INDIA ANNA DRAVIDA MUNNETRA KAZHAGAM
33                                  BHARAT RASHTRA SAMITHI
60                                  BHARATIYA JANATA PARTY
462                PRESIDENT, ALL INDIA CONGRESS COMMITTEE
482                                               SHIVSENA
501                                     TELUGU DESAM PARTY
504      YSR CONGRESS PARTY (YUVAJANA SRAMIKA RYTHU CON...
640                        DRAVIDA MUNNETRA KAZHAGAM (DMK)
647                                 JANATA DAL ( SECULAR )
672         NATIONALIST CONGRESS PARTY MAHARASHTRA PRADESH
765                           ALL INDIA TRINAMOOL CONGRESS
1063                       BIHAR PRADESH JANTA DAL(UNITED)
1147                                   RASHTRIYA JANTA DAL
1157                                       AAM AADMI PARTY
1159                              ADYAKSHA SAMAJVADI PARTY
2228                                   SHIROMANI AKALI DAL
2697                                JHARKHAND MUKTI MORCHA
2967                  JAMMU AND KASHMIR NATIONAL CONFERENCE
4238                                       BIJU JANATA DAL
8777                                     GOA FORWARD PARTY
9768                         MAHARASHTRAWADI GOMNTAK PARTY
11546                             SIKKIM KRANTIKARI MORCHA
11933                                       JANASENA PARTY
18465                              SIKKIM DEMOCRATIC FRONT
Name: PartyName, dtype: object
```

```
# Convert the 'Date' column in both dataframes to pandas datetime format
donated['JournalDate'] = pd.to_datetime(donated['JournalDate'])
donated['PurchaseDate'] = pd.to_datetime(donated['PurchaseDate'])

received['DateEncashment'] = pd.to_datetime(received['DateEncashment'])

# Check for missing values in both dataframes
missing_values_donated = donated.isnull().sum()
missing_values_recieved = received.isnull().sum()

# Handle missing values in both dataframes (e.g., drop rows with missing values, impute missing values)
# ...

# Remove duplicate rows from both dataframes
donated.drop_duplicates(inplace=True)
received.drop_duplicates(inplace=True)
```

Data Cleaning

Data cleaning is the process of identifying and correcting errors, inconsistencies, and missing values in a dataset. It is an essential step in data preparation that ensures the accuracy and reliability of your analysis.

Steps for Data Cleaning:

1. **Identify Data Quality Issues:**
   - Check for missing values, duplicates, outliers, and inconsistencies.
   - Use descriptive statistics and visualizations to identify potential data quality issues.
2. **Handle Missing Values:**
   - Drop rows with missing values if they are not essential for your analysis.
   - Impute missing values using appropriate techniques, such as mean, median, or mode.
3. **Correct Errors and Inconsistencies:**
   - Correct any errors or inconsistencies in the data, such as typos, incorrect formatting, or inconsistent units.
   - Use data validation rules to ensure that the data is consistent and valid.
4. **Remove Duplicates:**
   - Identify and remove duplicate rows from the dataset.
   - Use unique identifiers or a combination of columns to identify duplicates.
5. **Validate and Verify:**
   - After cleaning the data, it is important to validate and verify the results.
   - Use data validation techniques to ensure that the cleaned data is accurate and consistent.
   - Perform additional checks to ensure that the cleaning process did not introduce any new errors.

```
# Identifying Missing Values
print("Missing Values in Donated Data:")
print(donated.isnull().sum())

print("\nMissing Values in Recieved Data:")
print(received.isnull().sum())

## Observation: Here we have observed that there is no missing values in both dataset of ours.
```

```
Missing Values in Donated Data:
SNo              0
Urn              0
JournalDate      0
PurchaseDate     0
ExpiryDate       0
Purchaser        0
Prefix           0
BondNumber       0
Denominations    0
PayBranchCode    0
PayTeller        0
dtype: int64

Missing Values in Recieved Data:
Sno               0
DateEncashment    0
PartyName         0
AccountNum        0
Prefix            0
BondNumber        0
Denominations     0
PayBranchCode     0
PayTeller         0
dtype: int64
```

```
# Identify duplicate rows in donated data
duplicate_donated = donated[donated.duplicated()]
duplicate_received = received[received.duplicated()]

print("Number of duplicate rows in donated data:", len(duplicate_donated))
print("Number of duplicate rows in received data:", len(duplicate_received))

# Observation -> Here we can observe that there is no duplicate of data, if there is presence of duplication we can remove that using drop_duplication() method
donated.drop_duplicates(inplace=True)
received.drop_duplicates(inplace=True)
```

```
Number of duplicate rows in donated data: 0
Number of duplicate rows in received data: 0
```

Exploring Datasets

- Highest donation donated by which donator?
- Highest donation received by which political party?
- Total Highest donation donated by which donator?
- Total Highest donation received by which political party?

- At which date Highest number of bond bought?
- At which date Highest number of bond encashed?
- Top 10 Purchaser
- Top 10 Receipient
- Total amount of electoral bond purchased each year
- Which is the most common denominations of electoral bond purchased
- Top fist Donor has donated in which political party most
-

```
#Highest donation donated by which donor?
highest_donation = donated['Denominations'].max()
highest_donor = donated[donated['Denominations'] == highest_donation]['Purchaser'].values[0]

def convert_to_words(amount):
  return num2words(amount, lang='en_IN')


highest_donation_in_words = convert_to_words(highest_donation)

print(f"Highest donation of {highest_donation_in_words} was donated by {highest_donor}")
```

```
Highest donation of one crore was donated by ESSEL MINING AND INDS LTD
```

```
# Highest donation donated by which receiver?
highest_received = received['Denominations'].max()
highest_receiver = received[received['Denominations'] == highest_received]['PartyName'].values[0]

def convert_to_words(amount):
  return num2words(amount, lang='en_IN')

highest_received_in_words = convert_to_words(highest_received)

print(f"Highest donation of {highest_received_in_words} was received by {highest_receiver}")
```

```
Highest donation of one crore was received by ALL INDIA ANNA DRAVIDA MUNNETRA KAZHAGAM
```

Double-click (or enter) to edit

```
#Total highest donation made and by which Purchaser

total_donated = donated.groupby('Purchaser')['Denominations'].sum().sort_values(ascending=False)
highest_donor = total_donated.index[0]
highest_donation = total_donated.iloc[0]

print(f"The highest total donation of {highest_donation} was made by {highest_donor}")

# Observation-> Future Gaming And Hotel Services has donated twelve billion eighty million INR, Highest donator
```

```
The highest total donation of 12080000000 was made by FUTURE GAMING AND HOTEL SERVICES PR
```

```
#Total Highest donation received by which PartyName

total_received = received.groupby('PartyName')['Denominations'].sum().sort_values(ascending=False)
highest_receiver = total_received.index[0]
highest_received_donation = total_received.iloc[0]

print(f"The highest total donation of {highest_received_donation} was received by {highest_receiver}")

# Observation-> BJP (Bhartiya Janta Party) has received sixty billion six hundred five million one hundred eleven thousand INR, Highest Receiver
```

```
The highest total donation of 60605111000 was received by BHARATIYA JANATA PARTY
```

```
#Total Highest donation donated at which date?

highest_donation_date = donated['Denominations'].idxmax()
highest_donation_date_string = donated.loc[highest_donation_date, 'PurchaseDate']

print(f"Highest donation was donated on {highest_donation_date_string}")
```

```
Highest donation was donated on 2019-04-12
```

```
#Total highest donation encashed at which date?

highest_received_date = received['Denominations'].idxmax()
highest_received_date_string = received.loc[highest_received_date, 'DateEncashment']

print(f"Highest donation was encashed on {highest_received_date_string}")
```

```
Highest donation was encashed on 2019-04-12 00:00:00
```

```
# Top 10 Purchaser

top_10_purchasers = donated.groupby('Purchaser')['Denominations'].sum().sort_values(ascending=False).head(10)

print("Top 10 Purchasers:")
print(top_10_purchasers)
```

```
Top 10 Purchasers:
Purchaser
FUTURE GAMING AND HOTEL SERVICES PR               12080000000
MEGHA ENGINEERING AND INFRASTRUCTURES LI MITED     8210000000
QWIKSUPPLYCHAINPRIVATELIMITED                      4100000000
HALDIA ENERGY LIMITED                              3770000000
VEDANTA LIMITED                                    3756500000
ESSEL MINING AND INDS LTD                          2245000000
WESTERN UP POWER TRANSMISSION COMPANY LI MITED     2200000000
KEVENTER FOODPARK INFRA LIMITED                    1950000000
MADANLAL LTD.                                      1855000000
BHARTI AIRTEL LIMITED                              1830000000
Name: Denominations, dtype: int64
```

```
# Top 10 receiver

top_10_receivers = received.groupby('PartyName')['Denominations'].sum().sort_values(ascending=False).head(10)

print("Top 10 Receivers:")
print(top_10_receivers)
```

```
Top 10 Receivers:
PartyName
BHARATIYA JANATA PARTY                                             60605111000
ALL INDIA TRINAMOOL CONGRESS                                       16095314000
PRESIDENT, ALL INDIA CONGRESS COMMITTEE                            14218655000
BHARAT RASHTRA SAMITHI                                             12147099000
BIJU JANATA DAL                                                     7755000000
DRAVIDA MUNNETRA KAZHAGAM (DMK)                                     6390000000
YSR CONGRESS PARTY (YUVAJANA SRAMIKA RYTHU CONGRESS PARTY)          3370000000
TELUGU DESAM PARTY                                                  2188800000
SHIVSENA                                                            1593814000
RASHTRIYA JANTA DAL                                                  735000000
Name: Denominations, dtype: int64
```

```
# Total amount of electoral bond purchased each year

donated['Year'] = pd.to_datetime(donated['PurchaseDate']).dt.year
total_purchased_by_year = donated.groupby('Year')['Denominations'].sum()
print("Total amount of electoral bond purchased each year:")
print(total_purchased_by_year)
```

```
Total amount of electoral bond purchased each year:
Year
2019    17661280000
2020     3639601000
2021    15022927000
2022    37048576000
2023    42464745000
2024     5718003000
Name: Denominations, dtype: int64
```

```
#Which is the most common denominations of electoral bond purchased

most_common_denomination = donated['Denominations'].value_counts().idxmax()
print(f"The most common denomination of electoral bond purchased is {num2words(most_common_denomination, lang='en_IN')}")
```

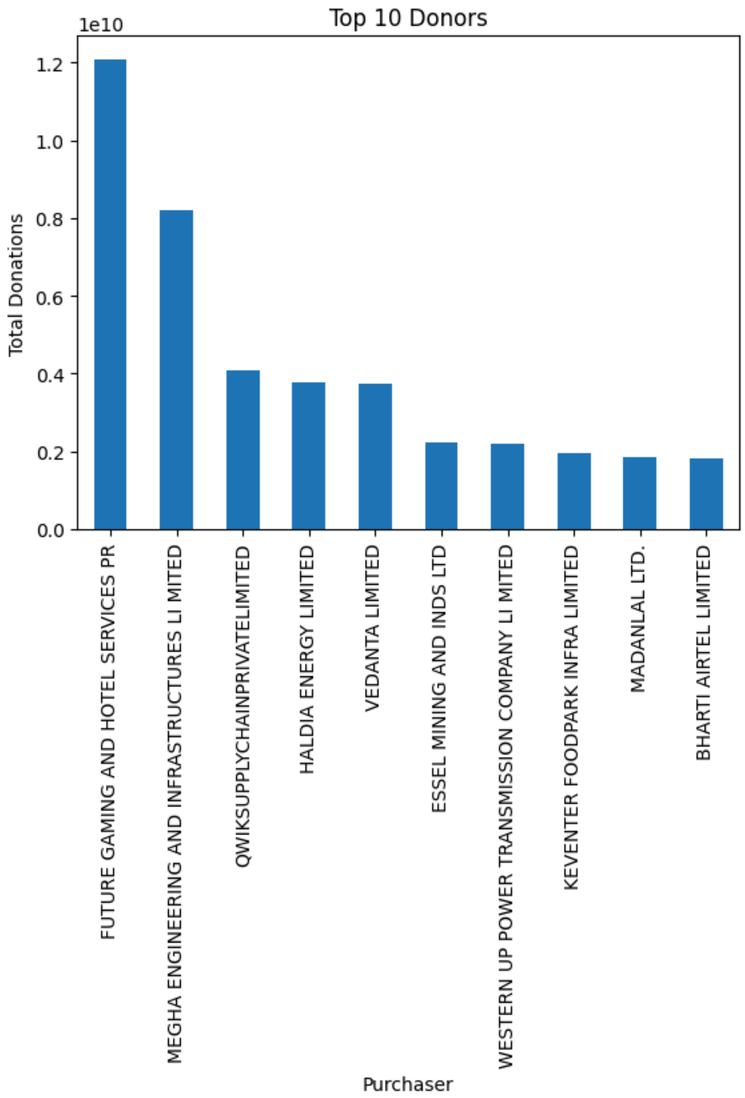
```
The most common denomination of electoral bond purchased is one crore
```

```
Start coding or generate with AI.
```

Data Visualization

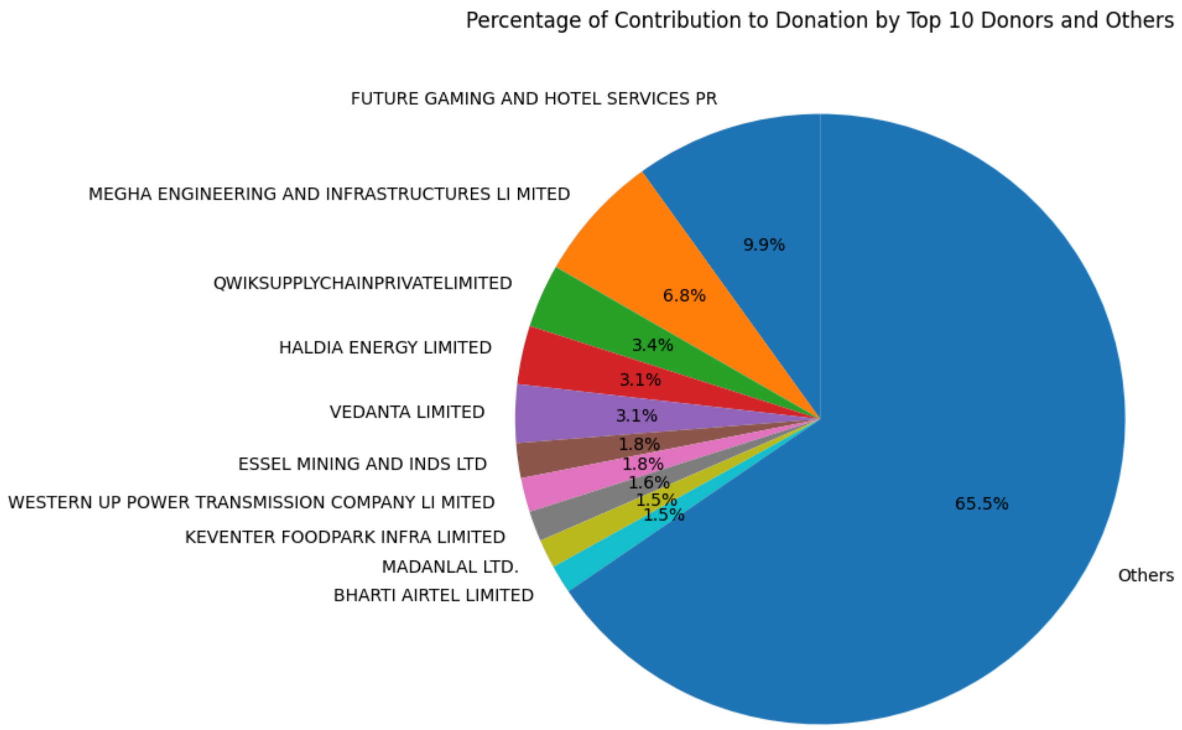
Data visualization is a critical aspect of data analysis that involves creating graphical representations of data to help convey information clearly and effectively. The primary goal is to turn complex data sets into visual insights that are easier to understand and interpret.

```
# Implementing Bar Chart to Visulize the total amount donated by each top 10 donor

donated.groupby('Purchaser')['Denominations'].sum().sort_values(ascending=False).head(10).plot(kind='bar')

plt.xlabel("Purchaser")
plt.ylabel("Total Donations")
plt.title("Top 10 Donors")
plt.show()
```

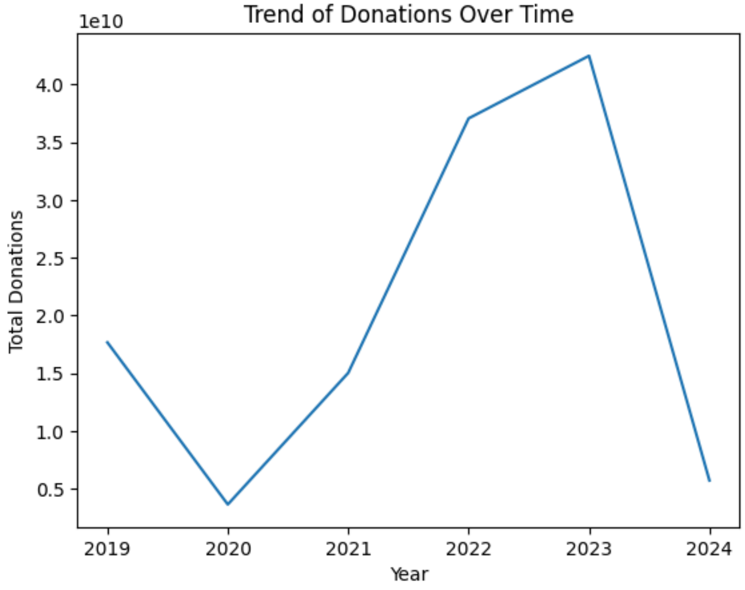


Double-click (or enter) to edit

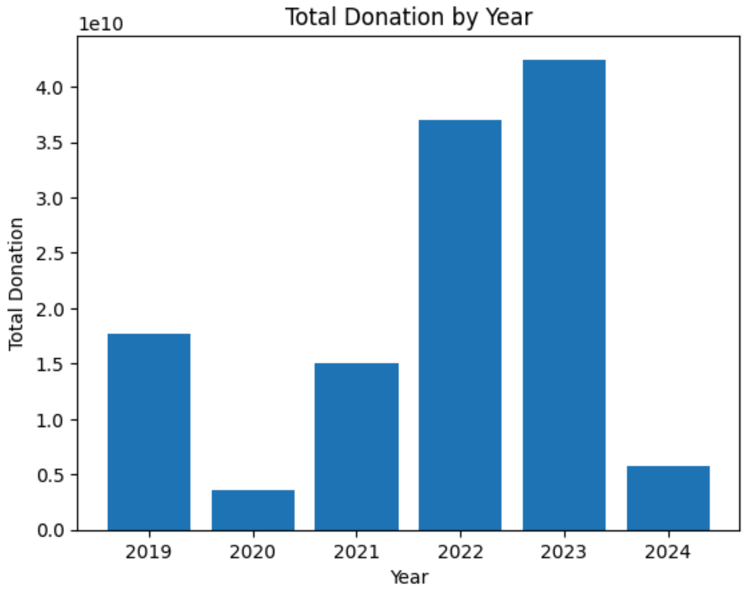
```
# Pie chart to show percentage of contribution to donation by each donator top 10 donators, and other donators as other and total summation of their percentage

donated_top_10 = donated.groupby('Purchaser')['Denominations'].sum().sort_values(ascending=False).head(10)
others = donated[~donated['Purchaser'].isin(donated_top_10.index)]['Denominations'].sum()
donated_top_10['Others'] = others
donated_top_10_percentage = donated_top_10 / donated['Denominations'].sum() * 100

plt.figure(figsize=(15, 8))
plt.pie(donated_top_10_percentage, labels=donated_top_10.index, autopct="%1.1f%%", startangle=90)
plt.title('Percentage of Contribution to Donation by Top 10 Donors and Others')
plt.show()
```



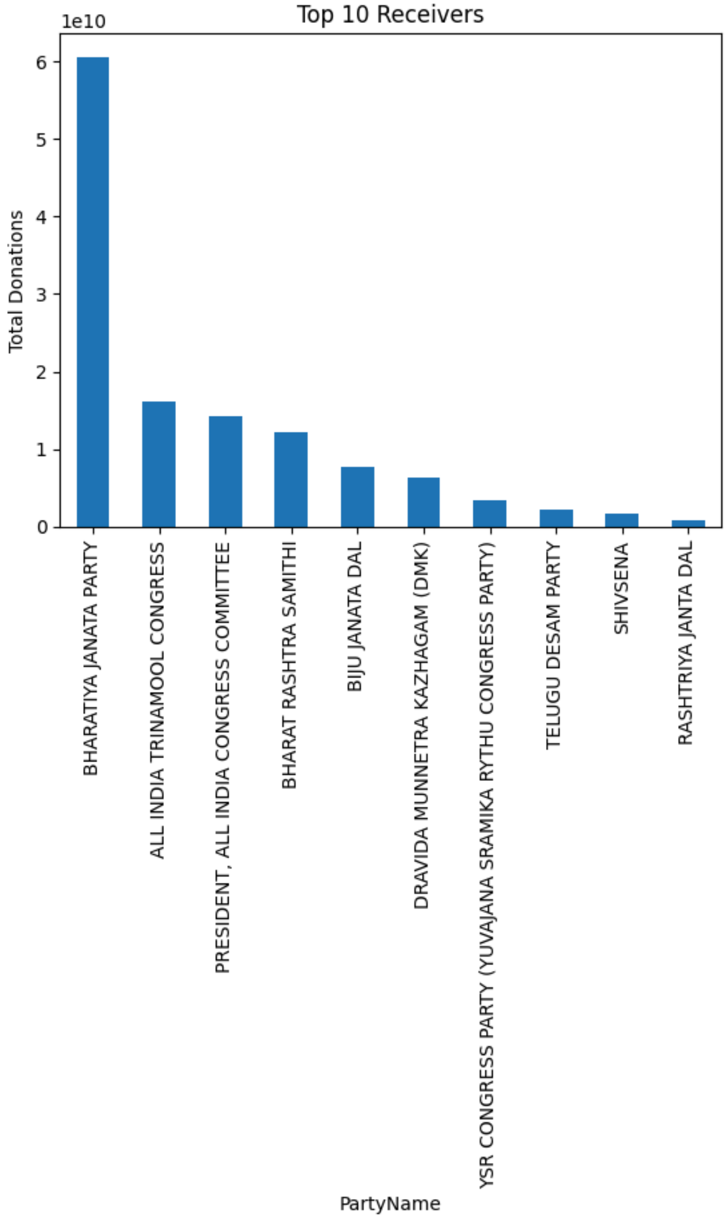
```
# Line chart To visualize the trend of donations over time.

donated_by_year = donated.groupby('Year')['Denominations'].sum()

plt.plot(donated_by_year.index, donated_by_year.values)
plt.xlabel('Year')
plt.ylabel('Total Donations')
plt.title('Trend of Donations Over Time')
plt.show()
```



```
donated['Year'] = pd.to_datetime(donated['PurchaseDate']).dt.year
total_donated_by_year = donated.groupby('Year')['Denominations'].sum()

plt.bar(total_donated_by_year.index, total_donated_by_year.values)

plt.xlabel("Year")
plt.ylabel("Total Donation")
plt.title("Total Donation by Year")

plt.show()
```



```
# Top 10 Political party encashed donation
received.groupby('PartyName')['Denominations'].sum().sort_values(ascending=False).head(10).plot(kind='bar')

plt.xlabel("PartyName")
plt.ylabel("Total Donations")
plt.title("Top 10 Receivers")
plt.show()
```



```
# Distribution of Denomination encashment of Top 7 Political party and others
grouped_data = received.groupby('PartyName')['Denominations'].sum()
top_7_parties = grouped_data.sort_values(ascending=False).head(7)
others = grouped_data[~grouped_data.index.isin(top_7_parties.index)].sum()
top_7_and_others = pd.DataFrame({
    'PartyName': ['Others'] + list(top_7_parties.index),
    'Denominations': [others] + list(top_7_parties.values)
})
```

```
})

plt.figure(figsize=(15, 8))
```