<u>Problem:</u> 1)

Your Friend has developed the Product and he wants to establish the product startup and he is searching for a perfect location where getting the investment has a high chance. But due to its financial restriction, he can choose only between three locations - Bangalore, Mumbai, and NCR. As a friend, you want to help your friend deciding the location. NCR include Gurgaon, Noida and New Delhi. Find the location where the most number of funding is done. That means, find the location where startups has received funding maximum number of times. Plot the bar graph between location and number of funding. Take city name "Delhi" as "New Delhi". Check the case-sensitiveness of cities also. That means, at some place instead of "Bangalore", "bangalore" is given. Take city name as "Bangalore". For few startups multiple locations are given, one Indian and one Foreign. Consider the startup if any one of the city lies in given locations.

Explanation:

The code for above problem performs the following tasks:
1. Imports the necessary libraries: `numpy`, `pandas`, and `matplotlib.pyplot`.
2. Reads a CSV file named 'startup_funding.csv' into a pandas DataFrame called `data`.
3. Drops rows with missing values in the 'CityLocation' column using `dropna()`.
4. Defines a function named `city(s)` that takes a string `s` representing a city location. The function splits the string by '/' if present, checks if either part of the split contains one of the desired cities ('Bangalore', 'Mumbai', 'Gurgaon', 'Noida', 'New Delhi'), and returns the corresponding city name after removing any leading or trailing whitespace.
5. Applies the `city()` function to the 'CityLocation' column of the DataFrame using `apply()`, assigning the modified values back to the 'CityLocation' column.
6. Performs multiple replacements in the 'CityLocation' column using `replace()`. These replacements convert 'Delhi' to 'New Delhi', 'bangalore' to 'Bangalore', 'New Delhi' to 'NCR', 'Gurgaon' to 'NCR', and 'Noida' to 'NCR' for standardization purposes.
7. Filters the DataFrame to include only rows where the 'CityLocation' is 'Bangalore', 'Mumbai', or 'NCR'.
8. Calculates the value counts of each city in the 'CityLocation' column using `value_counts()`, resulting in a Series with city names as index and corresponding counts as values.
9. Assigns the index of the value counts Series to the variable `city` and the values to the variable `startup`.
10. Creates a bar plot using `plt.bar()` to visualize the number of fundings received in each city.
11. Adds labels and a title to the plot.
12. Displays the plot using `plt.show()`.
13. Iterates over the range of the number of cities using `city.shape[0]` and prints the city name and the corresponding count of fundings received.

The code aims to analyze the distribution of fundings received by different cities, specifically focusing on 'Bangalore', 'Mumbai', and 'NCR' (New Delhi National Capital Region). The bar plot visualizes the number of fundings received by each city, with the x-axis representing the city names and the y-axis representing the count of fundings. The loop at the end prints the city names and the corresponding count of fundings received for further examination.

<u>Problem:</u> 2)

Even after trying for so many times, your friend’s startup could not find the investment. So you decided to take this matter in your hand and try to find the list of investors who probably can invest in your friend’s startup. Your list will increase the chance of your friend startup getting some initial investment by contacting these investors. Find the top 5 investors who have invested maximum number of times (consider repeat investments in one company also). In a startup, multiple investors might have invested. So consider each investor for that startup. Ignore undisclosed investors.

Explanation:

The code for above problem performs the following tasks:
1. Imports the necessary libraries: `pandas` and `matplotlib.pyplot`.
2. Reads a CSV file named 'startup_funding.csv' into a pandas DataFrame called `file`.
3. Drops rows with missing values in the 'InvestorsName' column using `dropna()`.
4. Initializes an empty dictionary called `dic` to store the count of investments for each investor.
5. Iterates over each value in the 'InvestorsName' column of the DataFrame.
6. Converts the investor name to a string and checks if it contains a comma (',').
7. If there is a comma, the name is split by ',' to handle multiple investors in a single entry.
8. For each individual investor in the split result, the code checks if the investor name exists in the dictionary `dic`. If it does, the count is incremented by 1; otherwise, a new entry is added with a count of 1.
9. If there is no comma in the name, the code follows the same process as above to handle single investor entries.
10. Creates a pandas Series called `df` using the values and keys from the `dic` dictionary. The values become the data, and the keys become the index.
11. Sorts the `df` Series in descending order and keeps only the top five entries.
12. Creates a bar plot using `plt.bar()` to visualize the number of investments for each investor.
13. Sets the rotation of x-axis labels to 15 degrees using `plt.xticks(rotation=15)` for better readability.
14. Adds labels and a title to the plot.
15. Displays the plot using `plt.show()`.
16. Prints the top five investors and their respective investment counts in a loop.

The code aims to analyze the investment data for different investors in the startup funding dataset. The bar plot shows the number of investments made by each investor, highlighting the top five investors. The loop at the end prints the names of the top five investors and their corresponding investment counts for further examination.

Problem: 3)

After re-analysing the dataset you found out that some investors have invested in the same startup at different number of funding rounds. So before finalising the previous list, you want to improvise it by finding the top 5 investors who have invested in different number of startups. This list will be more helpful than your previous list in finding the investment for your friend startup. Find the top 5 investors who have invested maximum number of times in different companies. That means, if one investor has invested multiple times in one startup, count one for that company. There are many errors in startup names. Ignore correcting all, just handle the important ones - Ola, Flipkart, Oyo and Paytm.

Explanation:
The code for above problem the following tasks:
1. Imports the necessary libraries: \`pandas\` and \`matplotlib.pyplot\`.
2. Reads a CSV file named 'startup_funding.csv' into a pandas DataFrame called \`file\`.
3. Drops rows with missing values in the 'InvestorsName' and 'StartupName' columns using \`dropna()\`.
4. Performs multiple replacements in the 'StartupName' column using \`replace()\`. These replacements standardize the names of certain startups (e.g., 'Ola Cabs' to 'Ola', 'Flipkart.com' to 'Flipkart') to ensure consistency.
5. Initializes two empty lists, \`invename\` and \`starname\`, to store investor names and startup names, respectively.
6. Iterates over each row in the DataFrame using \`iterrows()\`.
7. Extracts the investor name and startup name from each row.
8. If there is a comma (',') in the investor name, it is split by ',' to handle multiple investors in a single entry.
9. For each individual investor in the split result, the code appends the investor name and the corresponding startup name to the \`invename\` and \`starname\` lists.
10. If there is no comma in the investor name, the code appends the investor name and startup name to the lists as separate entries.
11. Creates a new DataFrame called \`df\` using the \`invename\` and \`starname\` lists, with columns 'InvestorsName' and 'StartupName', respectively.
12. Filters out any rows where the 'InvestorsName' is an empty string.
13. Groups the DataFrame by 'InvestorsName' and counts the number of unique 'StartupName' entries for each investor using \`nunique()\`.
14. Sorts the resulting Series in descending order and keeps only the top five entries.
15. Creates a bar plot using \`plt.bar()\` to visualize the number of investments in different startups for each investor.
16. Sets the rotation of x-axis labels to 15 degrees using \`plt.xticks(rotation=15)\` for better readability.
17. Adds labels and a title to the plot.
18. Displays the plot using \`plt.show()\`.
19. Prints the top five investors and the corresponding number of unique startups they have invested in.

The code aims to analyze the investment patterns of different investors by counting the number of unique startups they have invested in. The bar plot shows the top five investors with the highest number of investments in different startups. The loop at the end prints the names of the top five investors and the number of unique startups they have invested in for further examination.

<u>Problem:</u> 4)

Even after putting so much effort in finding the probable investors, it didn't turn out to be helpful for your friend. So you went to your investor friend to understand the situation better and your investor friend explained to you about the different Investment Types and their features. This new information will be helpful in finding the right investor. Since your friend startup is at an early stage startup, the best-suited investment type would be - Seed Funding and Crowdfunding. Find the top 5 investors who have invested in a different number of startups and their investment type is Crowdfunding or Seed Funding. Correct spelling of investment types are - "Private Equity", "Seed Funding", "Debt Funding", and "Crowd Funding". Keep an eye for any spelling mistake. You can find this by printing unique values from this column. There are many errors in startup names. Ignore correcting all, just handle the important ones - Ola, Flipkart, Oyo and Paytm.

Explanation:

The code for above problem performs the following tasks:
1. Imports the necessary libraries: `pandas` and `matplotlib.pyplot`.
2. Reads a CSV file named 'startup_funding.csv' into a pandas DataFrame called `file`.
3. Drops rows with missing values in the 'InvestorsName', 'StartupName', and 'InvestmentType' columns using `dropna()`.
4. Performs multiple replacements in the 'InvestmentType' and 'StartupName' columns using `replace()`. These replacements standardize the investment types (e.g., 'SeedFunding' to 'Seed Funding', 'Crowd funding' to 'Crowd Funding') and startup names (e.g., 'Ola Cabs' to 'Ola', 'Flipkart.com' to 'Flipkart') to ensure consistency.
5. Filters the DataFrame to include only rows where the 'InvestmentType' is either 'Seed Funding' or 'Crowd Funding'.
6. Initializes two empty lists, `invename` and `starname`, to store investor names and startup names, respectively.
7. Iterates over each row in the DataFrame using `iterrows()`.
8. Extracts the investor name and startup name from each row.
9. If there is a comma (',') in the investor name, it is split by ',' to handle multiple investors in a single entry.
10. For each individual investor in the split result, the code appends the investor name and the corresponding startup name to the `invename` and `starname` lists.
11. If there is no comma in the investor name, the code appends the investor name and startup name to the lists as separate entries.
12. Creates a new DataFrame called `df` using the `invename` and `starname` lists, with columns 'InvestorsName' and 'StartupName', respectively.
13. Filters out any rows where the 'InvestorsName' is 'Undisclosed Investors', 'Undisclosed investors', or an empty string.
14. Groups the DataFrame by 'InvestorsName' and counts the number of unique 'StartupName' entries for each investor using `nunique()`.
15. Sorts the resulting Series in descending order and keeps only the top five entries.
16. Creates a bar plot using `plt.bar()` to visualize the number of investments in different startups for each investor.
17. Sets the rotation of x-axis labels to 15 degrees using `plt.xticks(rotation=15)` for better readability.
18. Adds labels and a title to the plot.

19. Displays the plot using `plt.show()`.
20. Prints the top five investors and the corresponding number of unique startups they have invested in.

The code aims to analyze the investment patterns of different investors by counting the number of unique startups they have invested in. It focuses on the specific investment types of 'Seed Funding' and 'Crowd Funding' and excludes any rows with undisclosed investors. The bar plot shows the top five investors with the highest number of investments in different startups within the specified investment types. The loop at the end prints the names of the top five investors and the number of unique startups they have invested in for further examination.

Problem: 5)

Due to your immense help, your friend startup successfully got seed funding and it is on the operational mode. Now your friend wants to expand his startup and he is looking for new investors for his startup. Now you again come as a saviour to help your friend and want to create a list of probable new new investors. Before moving forward you remember your investor friend advice that finding the investors by analysing the investment type. Since your friend startup is not in early phase it is in growth stage so the best-suited investment type is Private Equity. Find the top 5 investors who have invested in a different number of startups and their investment type is Private Equity. Correct spelling of investment types are - "Private Equity", "Seed Funding", "Debt Funding", and "Crowd Funding". Keep an eye for any spelling mistake. You can find this by printing unique values from this column. There are many errors in startup names. Ignore correcting all, just handle the important ones - Ola, Flipkart, Oyo and Paytm.

Explanation:

The code for above problem performs the following tasks:
1. Imports the necessary libraries: `pandas` and `matplotlib.pyplot`.
2. Reads a CSV file named 'startup_funding.csv' into a pandas DataFrame called `file`.
3. Drops rows with missing values in the 'InvestorsName', 'StartupName', and 'InvestmentType' columns using `dropna()`.
4. Performs multiple replacements in the 'InvestmentType' and 'StartupName' columns using `replace()`. These replacements standardize the investment types (e.g., 'SeedFunding' to 'Seed Funding', 'Crowd funding' to 'Crowd Funding') and startup names (e.g., 'Ola Cabs' to 'Ola', 'Flipkart.com' to 'Flipkart') to ensure consistency.
5. Filters the DataFrame to include only rows where the 'InvestmentType' is 'Private Equity'.
6. Initializes two empty lists, `invename` and `starname`, to store investor names and startup names, respectively.
7. Iterates over each row in the DataFrame using `iterrows()`.
8. Extracts the investor name and startup name from each row.
9. If there is a comma (',') in the investor name, it is split by ',' to handle multiple investors in a single entry.
10. For each individual investor in the split result, the code appends the investor name and the corresponding startup name to the `invename` and `starname` lists.
11. If there is no comma in the investor name, the code appends the investor name and startup name to the lists as separate entries.
12. Creates a new DataFrame called `df` using the `invename` and `starname` lists, with columns 'InvestorsName' and 'StartupName', respectively.

13. Filters out any rows where the 'InvestorsName' is 'Undisclosed Investors', 'Undisclosed investors', or an empty string.
14. Groups the DataFrame by 'InvestorsName' and counts the number of unique 'StartupName' entries for each investor using `nunique()`.
15. Sorts the resulting Series in descending order and keeps only the top five entries.
16. Creates a bar plot using `plt.bar()` to visualize the number of investments in different startups for each investor.
17. Sets the rotation of x-axis labels to 15 degrees using `plt.xticks(rotation=15)` for better readability.
18. Adds labels and a title to the plot.
19. Displays the plot using `plt.show()`.
20. Prints the top five investors and the corresponding number of unique startups they have invested in.

The code aims to analyze the investment patterns of different investors specifically for the investment type 'Private Equity'. It filters the data to include only rows with 'Private Equity' as the investment type and excludes rows with undisclosed investors. The bar plot shows the top five investors with the highest number of investments in different startups within the specified investment type. The loop at the end prints the names of the top five investors and the number of unique startups they have invested in for further examination.