Q1.

```
locations=np.array(['Bangalore','Mumbai','New Delhi','Noida','Gurgaon'])
city_fundings_count=[]
```

locations is a numpy array having cities which my friend can choose because of the financial restriction.

city_fundings_count is an array for storing number of fundings these cities (present in location) got.

```
df=pd.read_csv("startup_funding.csv")
```

df is a dataframe containing data of a csv file read using function read_csv.

```
df.CityLocation.fillna('', inplace=True)
```

Next, I used fillna function to fill NaN values with emty string as we cannot use dropna because all the row have NaN values somewhere due to which dropna will drop all the rows.

```
df.CityLocation.replace('bangalore','Bangalore', inplace=True)
df.CityLocation.replace('Delhi','New Delhi', inplace=True)
```

using replace function in order to give correct names of the cities

```
def indian_city(s):
    for i in s.split('/'):
        city=i.strip()
        if city in locations:
            return city
    return s
df.CityLocation=df.CityLocation.apply(indian_city)
```

using apply to extract Indian cities from some CityLocations which have one indian and one foreign citiy mentioned within them.

So, apply is calling a func Indian city in which we are splitting the string from '/' using split func which will return a list having 2 elements(cities), so in that list , we are checking for both the elements(cities) ,if that city is in the location array, we return that city and if both the cities are not in the location array, we return the string as it is only.

```
s=df.CityLocation[df.CityLocation!=''].value_counts()
```

now, we are counting the number of fundings by using value_counts which counts the number of occurrences of the values in a column, in this case it counts the number of occurrences of cities in the CityLocation column and storing it into ‘s’

```
for i in locations:
    city_fundings_count.append(s.loc[i])
np_city_fundings_count=np.array(city_fundings_count)
```

appending number of fundings counted above into ‘city_fundings_count’ in same order of cities as in the ‘location’

then, making numpy array of ‘city_fundings_count’

```
best_location=locations[np.where(np_city_fundings_count==max(np_city_fundings_count))][0]
```

now getting the best location in location nparray using np.where where np_city_fundings_count is maximum, which returns a list, so in order to get the first element we do [0]

at last, plotting graph and printing best location

Q2.

```
dictionary={}                                   #dictionary
df=pd.read_csv('startup_funding.csv')       #reading csv
df.InvestorsName.fillna('', inplace=True)
```

used fillna function to fill NaN values with emty string as we cannot use dropna because all the row have NaN values somewhere due to which dropna will drop all the rows.

```
def investorNames(arr):
    d={}
    for string in arr:
        if ',' in string:
            for i in string.split(','):
                a=i.strip()
                d[a]= d.get(a,0)+1
        else:
```

```
        d[string]= d.get(string,0)+1
    return d
```

invsetorNames function with one argument, for loop to count investors, in this we check each entry in arr, if the entry has ‘,’ in it, it means it has multiple investors, so we split it from ‘,’ which returns a list, and for elements in that list we count and add that to the dict d and if the entry don’t have ‘,’ in it this means it contains only one investor, so we count if it is present in the dict or add it to the dict if not present and the function returns the dict.

```
dictionary=investorNames(df.InvestorsName)
df1= pd.DataFrame(list(dictionary.values()), list(dictionary.keys()))
```

calling function investorNames and passing InvestorsNames column in as parameter which returns a dict

making a dataframe df1 with investors name as index and count as column with name [0]

```
s=df1[df1.index!=''].sort_values([0],ascending=False)[:5]
for i in s.index:
    print(i)
```

now, df1 where df1 is not equal to empty string , we sort values in descending order and pickup top five values from it using [:5] and store it in ‘s’ and in s.index array, we get the names of top 5 investors

Q3.

```
dictionary ={}                                  #dictionary d
df=pd.read_csv('startup_funding.csv')       #reading csv
df.InvestorsName.fillna('', inplace=True)
```

used fillna function to fill NaN values with emty string as we cannot use dropna because all the row have NaN values somewhere due to which dropna will drop all the rows.

```
df.StartupName.replace('Ola Cabs','Ola', inplace=True)
df.StartupName.replace('Olacabs','Ola', inplace=True)
df.StartupName.replace('OyoRooms','Oyo',inplace=True)
df.StartupName.replace('Oyo Rooms','Oyo',inplace=True)
df.StartupName.replace('Oyorooms','Oyo',inplace=True)
df.StartupName.replace('OYO Rooms','Oyo',inplace=True)
```

```
df.StartupName.replace('Paytm Marketplace','Paytm',inplace=True)

df.StartupName.replace('Flipkart.com','Flipkart', inplace=True)
```

using replace to give same spelling to the startups which are same but appearing with different spellings

```
df['Grouped']=df.groupby(df.StartupName).InvestorsName.transform(lambda x: ','.join(x)).values

df1=df[['StartupName','Grouped']].drop_duplicates()
```

using groupby to group df by StartupName and picking InvestorsName having the same startup, using transform and join, these investor names are joined by comma into a single string and the string is added to the new column 'Grouped' in the rows where the StartupName is that particular startup

then, new dataframe df1 with only 2 columns 'StartupName' and 'Grouped' of df after removing the duplicate rows

```
def  investorNames(arr):
    d={}
    for string in arr:
        l=[]
        if ',' in string:
            for i in string.split(','):
                a=i.strip()
                if a not in l:
                    d[a]= d.get(a,0)+1
                    l.append(a)
        else:
            d[string]= d.get(string,0)+1
    return d
```

Function definition of InvestorNames(arr), which checks the strings present in arr, if the string has comma(",") in it, it splits the string, strips out whitespaces (from end and front) from the substrings one by one and if the substring is not present in "l" array, it either adds the string(investor name) if not present in the dictionary "d" or increases its count by 1 if already present, and appends the string to the array 'l', and if it is already present in the array, it skips out the count(as we have to count investors name 1 time only) and if the string does not have comma in it, it just counts the name. After all strings in arr is checked and counted, function returns the dictionary d

```
dictionary=investorNames(df1.Grouped)
```

function investorNames is called and “Grouped” column of df1 is passed as arr and the value returned by the func is stored in the dictionary named ”dictionary”

```
df2= pd.DataFrame(list(dictionary.values()), list(dictionary.keys()))
s=df2[df2.index!=''].sort_values([0],ascending=False)[:5]
```

dataframe df2 is created with investor names as index (accessed using dictionary.values()), and column having counts(accessed using dictionary.keys())

then, df2 where index of df2 is not equal to empty string ““ is sorted by its [0] column in descending order using sort_values and its top 5 values are stored in variable s

```
for i in s.index:
    print(i)
```

now, investors names are printed from s.index

Q4.

```
dictionary ={}                                          #dictionary d
df=pd.read_csv('startup_funding.csv')       #reading csv
df.InvestorsName.fillna('', inplace=True)
```

used fillna function to fill NaN values with emty string as we cannot use dropna because all the row have NaN values somewhere due to which dropna will drop all the rows.

```
df.StartupName.replace('Ola Cabs','Ola', inplace=True)
df.StartupName.replace('Olacabs','Ola', inplace=True)
df.StartupName.replace('OyoRooms','Oyo',inplace=True)
df.StartupName.replace('Oyo Rooms','Oyo',inplace=True)
df.StartupName.replace('Oyorooms','Oyo',inplace=True)
df.StartupName.replace('OYO Rooms','Oyo',inplace=True)
df.StartupName.replace('Paytm Marketplace','Paytm',inplace=True)
df.StartupName.replace('Flipkart.com','Flipkart', inplace=True)
```

using replace to give same spelling to the startups which are same but appearing with different spellings

```
df.InvestmentType.replace('PrivateEquity','Private Equity', inplace=True)

df.InvestmentType.replace('Crowd funding','Crowd Funding', inplace=True)

df.InvestmentType.replace('SeedFunding','Seed Funding', inplace=True)

df.InvestmentType.fillna('',inplace=True)
```

using replace to give same spelling to the Investment type which are same but appearing with different spellings and InvestmentType with NaN values are filled with empty string

```
df1=df[(df.InvestmentType=='Crowd Funding') | (df.InvestmentType=='Seed Funding')]

df1['Grouped']=df1.groupby(df1.StartupName).InvestorsName.transform(lambda x: ','.join(x)).values

df1=df1[['StartupName','Grouped']].drop_duplicates()
```

df where investment type is either “Crowd Funding” or “Seed Funding” is stored in df1

using groupby on df1, grouping df1 by StartupName and picking their InvestorName and using transform and join, joining these names by comma and storing it in the new column “Grouped” in the rows where the StartupName is that particular startup

then, dataframe df1 with only 2 columns ‘StartupName’ and ‘Grouped’ of df1 after removing the duplicate rows

```
def  investorNames(arr):
    d={}
    for string in arr:
        l=[]
        if ',' in string:
            for i in string.split(','):
                a=i.strip()
                if a not in l:
                    d[a]= d.get(a,0)+1
                    l.append(a)
        else:
            d[string]= d.get(string,0)+1
    return d
```

Function definition of InvestorNames(arr), which checks the strings present in arr, if the string has comma(“,”) in it, it splits the string, strips out whitespaces for the substrings one by one and if the substring is not present in l array, it either adds the string(investor name) if not present in the dictionary “d” or increases its count by 1 if already present, and appends the string to the array ‘l’, and if it is already present in the array, it skips out the count(as we have to count investors name 1

time only) and if the string doesnot have comma in it, it just counts the name. After all strings in arr is checked and counted, function returns the dictionary d

```
dictionary=investorNames(df1.Grouped)
```

function investorNames is called and “Grouped” column of df1 is passed as arr and the value returned by the func is stored in the dictionary named ”dictionary”

```
df2= pd.DataFrame(list(dictionary.values()), list(dictionary.keys()))
s=df2[(df2.index!='') & (~df2.index.str.contains('Undisclosed'))].sort_values([0],ascending=False)[:5]
```

dataframe df2 is created with investor names as index (accessed using dictionary.values()), and column having counts(accessed using dictionary.keys())

then, df2 where index of df2 is not equal to empty string (“”) and index of df2 do not contain the word “Undisclosed” in it , that df2 is sorted by its [0] column in descending order using sort_values and its top 5 values are stored in variable s

```
for i in s.index:
    print(i)
```

now, investors names are printed from s.index

Q5.

```
dictionary ={}                                          #dictionary d
df=pd.read_csv('startup_funding.csv')       #reading csv
df.InvestorsName.fillna('', inplace=True)
```

used fillna function to fill NaN values with emty string as we cannot use dropna because all the row have NaN values somewhere due to which dropna will drop all the rows.

```
df.StartupName.replace('Ola Cabs','Ola', inplace=True)
df.StartupName.replace('Olacabs','Ola', inplace=True)
df.StartupName.replace('OyoRooms','Oyo',inplace=True)
df.StartupName.replace('Oyo Rooms','Oyo',inplace=True)
df.StartupName.replace('Oyorooms','Oyo',inplace=True)
df.StartupName.replace('OYO Rooms','Oyo',inplace=True)
df.StartupName.replace('Paytm Marketplace','Paytm',inplace=True)
```

```
df.StartupName.replace('Flipkart.com','Flipkart', inplace=True)
```

using replace to give same spelling to the startups which are same but appearing with different spellings

```
df.InvestmentType.replace('PrivateEquity','Private Equity', inplace=True)

df.InvestmentType.replace('Crowd funding','Crowd Funding', inplace=True)

df.InvestmentType.replace('SeedFunding','Seed Funding', inplace=True)

df.InvestmentType.fillna('',inplace=True)
```

using replace to give same spelling to the Investment type which are same but appearing with different spellings and InvestmentType with NaN values are filled with empty string

```
df1=df[(df.InvestmentType=='Private Equity')]

df1['Grouped']=df1.groupby(df1.StartupName).InvestorsName.transform(lambda x: ','.join(x)).values

df1=df1[['StartupName','Grouped']].drop_duplicates()
```

df where investment type is 'Private Equity' is stored in df1

using groupby on df1, grouping df1 by StartupName and picking their InvestorName and using transform and join, joining these names by comma and storing it in the new column “Grouped” in the rows where the StartupName is that particular startup

then, dataframe df1 with only 2 columns ‘StartupName’ and ‘Grouped’ of df1 after removing the duplicate rows

```
def  investorNames(arr):
    d={}
    for string in arr:
        l=[]
        if ',' in string:
            for i in string.split(','):
                a=i.strip()
                if a not in l:
                    d[a]= d.get(a,0)+1
                    l.append(a)
        else:
            d[string]= d.get(string,0)+1
    return d
```

Function definition of InvestorNames(arr), which checks the strings present in arr, if the string has comma(“,”) in it, it splits the string, strips out whitespaces for the substrings one by one and if the substring is not present in l array, it either adds the string(investor name) if not present in the dictionary “d” or increases its count by 1 if already present, and appends the string to the array ‘l’, and if it is already present in the array, it skips out the count(as we have to count investors name 1 time only) and if the string doesnot have comma in it, it just counts the name. After all strings in arr is checked and counted, function returns the dictionary d

```
dictionary=investorNames(df1.Grouped)
```

function investorNames is called and “Grouped” column of df1 is passed as arr and the value returned by the func is stored in the dictionary named ”dictionary”

```
df2= pd.DataFrame(list(dictionary.values()), list(dictionary.keys()))

s=df2[df2.index!=''].sort_values([0],ascending=False)[:5]
```

dataframe df2 is created with investor names as index (accessed using dictionary.values()), and column having counts(accessed using dictionary.keys())

then, df2 where index of df2 is not equal to empty string (“”) , that df2 is sorted by its [0] column in descending order using sort_values and its top 5 values are stored in variable s

```
for i in s.index:
    print(i)
```

now, investors names are printed from s.index