TASK 1 Visualising the population distribution

In [6]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [7]:
```
df = pd.read_csv("countries-table.csv")
```

In [8]:
```
df
```

Out[8]:

	country	rank	area	landAreaKm	cca2	cca3	netChange	growthRate	worldPercentage	density	densityMi	place	pop1980	pop2000	pop2010	pop2022	po
0	India	1	3287590.00	2973190.00	IN	IND	0.4184	0.0081	0.1785	480.5033	1244.5036	356	696828385	1059633675	1240613620	1417173173	1429
1	China	2	9706961.00	9424702.90	CN	CHN	-0.0113	-0.0002	0.1781	151.2696	391.7884	156	982372466	1264099069	1348191368	1425887337	1429
2	United States	3	9372610.00	9147420.00	US	USA	0.0581	0.0050	0.0425	37.1686	96.2666	840	223140018	282398554	311182845	338289857	339
3	Indonesia	4	1904569.00	1877519.00	ID	IDN	0.0727	0.0074	0.0347	147.8196	382.8528	360	148177096	214072421	244016173	275501339	277
4	Pakistan	5	881912.00	770880.00	PK	PAK	0.1495	0.0198	0.0300	311.9625	807.9829	586	80624057	154369924	194454498	235824862	240
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
229	Montserrat	230	102.00	102.00	MS	MSR	NaN	-0.0009	NaN	43.0000	111.3700	500	11452	5138	4938	4390	
230	Falkland Islands	231	12173.00	12173.00	FK	FLK	NaN	0.0029	NaN	0.3114	0.8066	238	2240	3080	3187	3780	
231	Niue	232	261.00	261.00	NU	NIU	0.0000	0.0005	NaN	7.4138	19.2017	570	3637	2074	1812	1934	
232	Tokelau	233	12.00	10.00	TK	TKL	NaN	0.0118	NaN	189.3000	490.2870	772	1647	1666	1367	1871	
233	Vatican City	234	0.44	0.44	VA	VAT	NaN	0.0157	NaN	1177.2727	3049.1364	336	733	651	596	510	

234 rows × 19 columns

In [9]:
```
df.shape
```

Out[9]:
```
(234, 19)
```

Checking columns in Dataset

In [10]:
```
df.columns
```

Out[10]:
```
Index(['country', 'rank', 'area', 'landAreaKm', 'cca2', 'cca3', 'netChange',
       'growthRate', 'worldPercentage', 'density', 'densityMi', 'place',
       'pop1980', 'pop2000', 'pop2010', 'pop2022', 'pop2023', 'pop2030',
       'pop2050'],
      dtype='object')
```

Information About The DataSet

In [11]:
```
df.info()
```

Out[11]:
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 234 entries, 0 to 233
Data columns (total 19 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   country         234 non-null    object 
 1   rank            234 non-null    int64  
 2   area            234 non-null    float64
 3   landAreaKm      234 non-null    float64
 4   cca2            233 non-null    object 
 5   cca3            234 non-null    object 
 6   netChange       226 non-null    float64
 7   growthRate      234 non-null    float64
 8   worldPercentage 228 non-null    float64
 9   density         234 non-null    float64
 10  densityMi       234 non-null    float64
 11  place           234 non-null    int64  
 12  pop1980         234 non-null    int64  
 13  pop2000         234 non-null    int64  
 14  pop2010         234 non-null    int64  
 15  pop2022         234 non-null    int64  
 16  pop2023         234 non-null    int64  
 17  pop2030         234 non-null    int64  
 18  pop2050         234 non-null    int64  
dtypes: float64(7), int64(9), object(3)
memory usage: 34.9+ KB
```

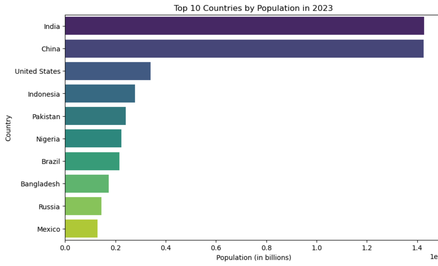
In [12]:
```
df.describe()
```

Out[12]:

	rank	area	landAreaKm	netChange	growthRate	worldPercentage	density	densityMi	place	pop1980	pop2000	pop2010	pop202
count	234.000000	2.340000e+02	2.340000e+02	226.000000	234.000000	228.000000	234.000000	234.000000	234.000000	2.340000e+02	2.340000e+02	2.340000e+02	2.340000e+0
mean	117.500000	5.814500e+05	5.571123e+05	0.010306	0.009737	0.004407	451.288182	1168.836388	439.085470	1.899462e+07	2.626947e+07	2.984524e+07	3.407441e+0
std	67.694165	1.761841e+06	1.689972e+06	0.034774	0.012350	0.017375	1979.362419	5126.548664	253.295484	8.178519e+07	1.116982e+08	1.242185e+08	1.367664e+0
min	1.000000	4.400000e-01	4.400000e-01	-0.028600	-0.074500	0.000000	0.138000	0.357400	4.000000	7.330000e+02	6.510000e+02	5.960000e+02	5.100000e+0
25%	59.250000	2.650000e+03	2.625875e+03	0.000000	0.002325	0.000100	39.747650	102.946450	223.000000	2.296142e+05	3.272420e+05	3.931490e+05	4.197385e+0
50%	117.500000	8.119950e+04	7.568925e+04	0.000900	0.008200	0.000750	97.481000	252.475800	439.000000	3.141146e+06	4.292907e+06	4.942770e+06	5.559944e+0
75%	175.750000	4.304258e+05	4.047876e+05	0.008000	0.016850	0.002925	242.928650	629.185350	659.750000	9.826054e+06	1.576230e+07	1.915957e+07	2.247650e+0
max	234.000000	1.709824e+07	1.637687e+07	0.418400	0.049800	0.178500	21402.705200	55433.006400	894.000000	9.823725e+08	1.264099e+09	1.348191e+09	1.425887e+0

Extracting Top 10 Countries by population and presenting a Barchart to analyse it.

In [13]:
```
top10_countries = df.sort_values(by='pop2023',ascending=False).head(10)

# Create a bar chart
plt.figure(figsize=(10, 6))
sns.barplot(x='pop2023', y='country', data=top10_countries, palette='viridis')
plt.title('Top 10 Countries by Population in 2023')
plt.xlabel('Population (in billions)')
plt.ylabel('Country')
plt.show()
```

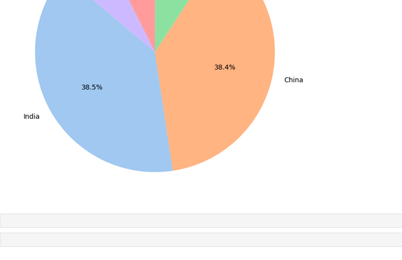


In [14]:
```
print(df.columns)
```

```
Index(['country', 'rank', 'area', 'landAreaKm', 'cca2', 'cca3', 'netChange',
       'growthRate', 'worldPercentage', 'density', 'densityMi', 'place',
       'pop1980', 'pop2000', 'pop2010', 'pop2022', 'pop2023', 'pop2030',
       'pop2050'],
      dtype='object')
```

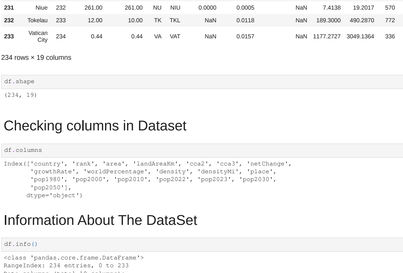
Extract data of population over time in India

In [15]:
```
# Select a specific country (e.g., 'India') for visualization
country_data = df[df['country'] == 'India']

# Extract population data over time
years = ['pop1980', 'pop2000', 'pop2010', 'pop2022', 'pop2023', 'pop2030', 'pop2050']
population = country_data[years].values.flatten()

# Plot a line graph
plt.figure(figsize=(10, 6))
plt.plot(years, population, marker='o', color='blue')
plt.title('Population Growth Over Time: India')
plt.xlabel('Year')
plt.ylabel('Population (in billions)')
plt.grid()
plt.show()
```



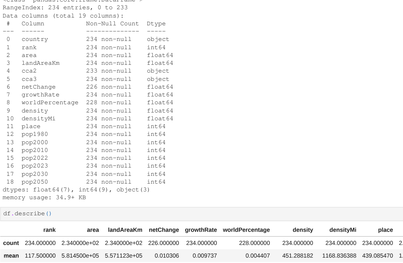
Distribution of population

In [16]:
```
plt.figure(figsize=(10, 6))
sns.histplot(df['growthRate'], bins=15, kde=True, color='green')
plt.title('Distribution of Population Growth Rates')
plt.xlabel('Growth Rate (%)')
plt.ylabel('Frequency')
plt.show()
```



Extracting Top 10 Countries by population and presenting a Barchart to analyse it.

In [17]:
```
# Sort data by world percentage and select the top 5
top5_countries = df.sort_values(by='worldPercentage', ascending=False).head(5)

# Plot a pie chart
plt.figure(figsize=(8, 8))
plt.pie(top5_countries['worldPercentage'], labels=top5_countries['country'], autopct='%1.1f%%', startangle=140, colors=sns.color_palette('pastel'))
plt.title('World Population Share by Top 5 Countries')
plt.show()
```



In [ ]:
```

```

In [ ]:
```

```