```
import pandas as pd
from tabulate import tabulate
df = pd.read_csv('Misje.csv')
df.head()
```

```
  CompanyName  SpaceBase     Time
RocketModel  \
0   RVSN USSR  Site 41/1  53:00.0  Molniya-M /Block ML | Molniya-1 n†-
133
1   RVSN USSR  Site 41/1  53:00.0  Molniya-M /Block ML | Molniya-1 n†-
133
2   RVSN USSR  Site 41/1  53:00.0  Molniya-M /Block ML | Molniya-1 n†-
133
3   RVSN USSR  Site 41/1  02:00.0      Molniya-M /Block 2BL | Cosmos
1977
4   RVSN USSR  Site 41/1  23:00.0      Molniya-M /Block 2BL | Cosmos
1974

    RocketStatus  RocketCost MissionStatus Country
Region  \
0  StatusRetired         NaN       Success  Russia  Plesetsk
Cosmodrome
1  StatusRetired         NaN       Success  Russia  Plesetsk
Cosmodrome
2  StatusRetired         NaN       Success  Russia  Plesetsk
Cosmodrome
3  StatusRetired         NaN       Success  Russia  Plesetsk
Cosmodrome
4  StatusRetired         NaN       Success  Russia  Plesetsk
Cosmodrome

         Date
0   8/12/1998
1   8/12/1998
2   8/12/1998
3  10/25/1988
4   10/3/1988
```

```
import pandas as pd



# Konwersja kolumny Time do formatu czasu
df['Time'] = pd.to_datetime(df['Time'], format='%H:%M.%S',
errors='coerce')

# Obliczenie liczby misji, sumy kosztów misji oraz stosunku sukcesu do
```

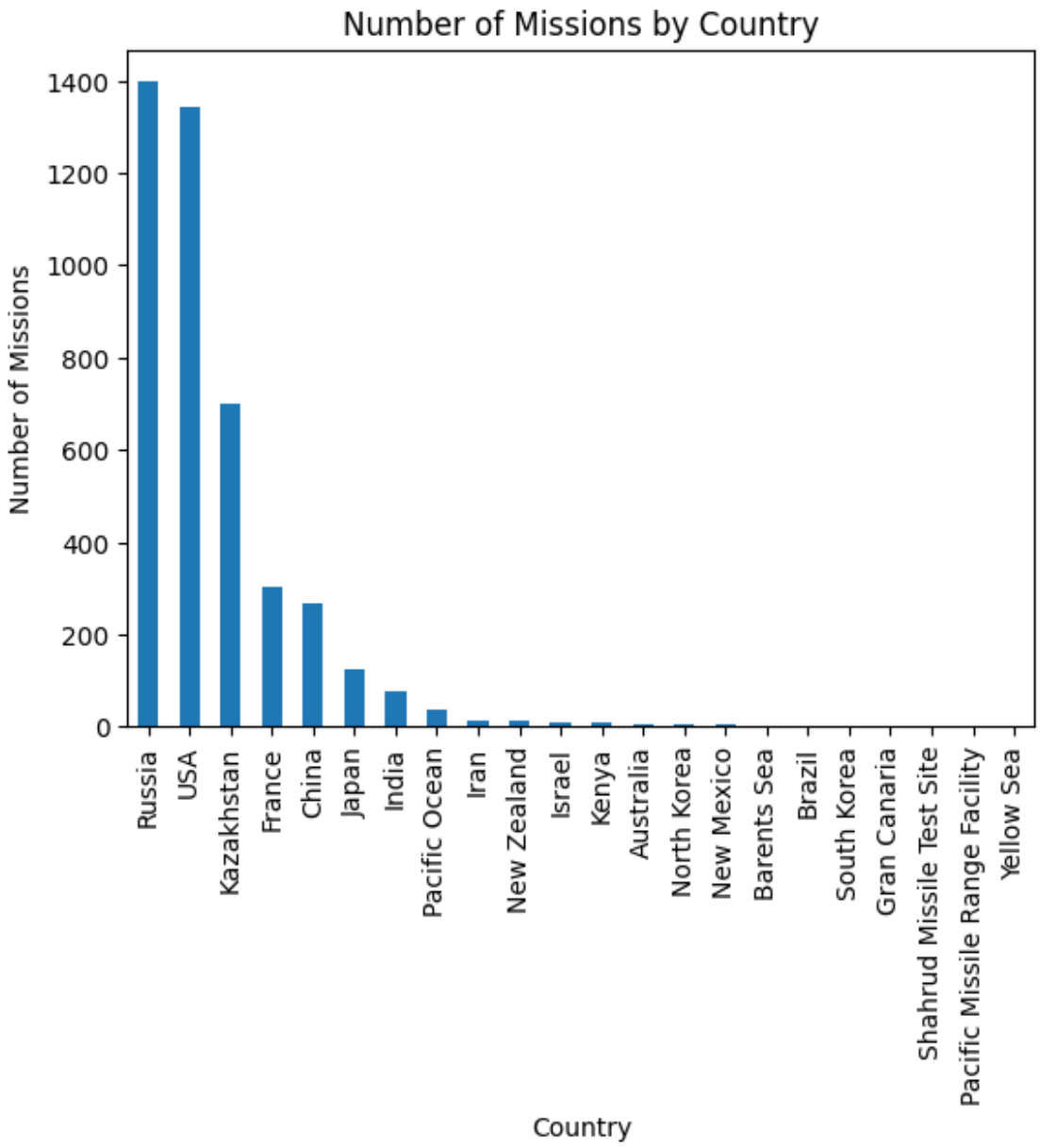
```
porażki
summary_by_country = df.groupby('Country').agg(
    NumberOfMissions=('MissionStatus', 'size'),
    TotalMissionCost=('RocketCost', 'sum'),
    SuccessFailureRatio=('MissionStatus', lambda x: (x ==
'Success').mean())
)

print(summary_by_country)
```

Country	NumberOfMissions	TotalMissionCost	\
Australia	6	0.00	
Barents Sea	3	0.00	
Brazil	3	0.00	
China	268	6363.26	
France	303	16285.00	
Gran Canaria	2	80.00	
India	76	2177.00	
Iran	13	0.00	
Israel	11	0.00	
Japan	126	3700.50	
Kazakhstan	701	12150.50	
Kenya	9	0.00	
New Mexico	4	0.00	
New Zealand	13	97.50	
North Korea	5	0.00	
Pacific Missile Range Facility	1	15.00	
Pacific Ocean	36	0.00	
Russia	1397	2189.30	
Shahrud Missile Test Site	1	0.00	
South Korea	3	0.00	
USA	1344	105192.32	
Yellow Sea	1	5.30	

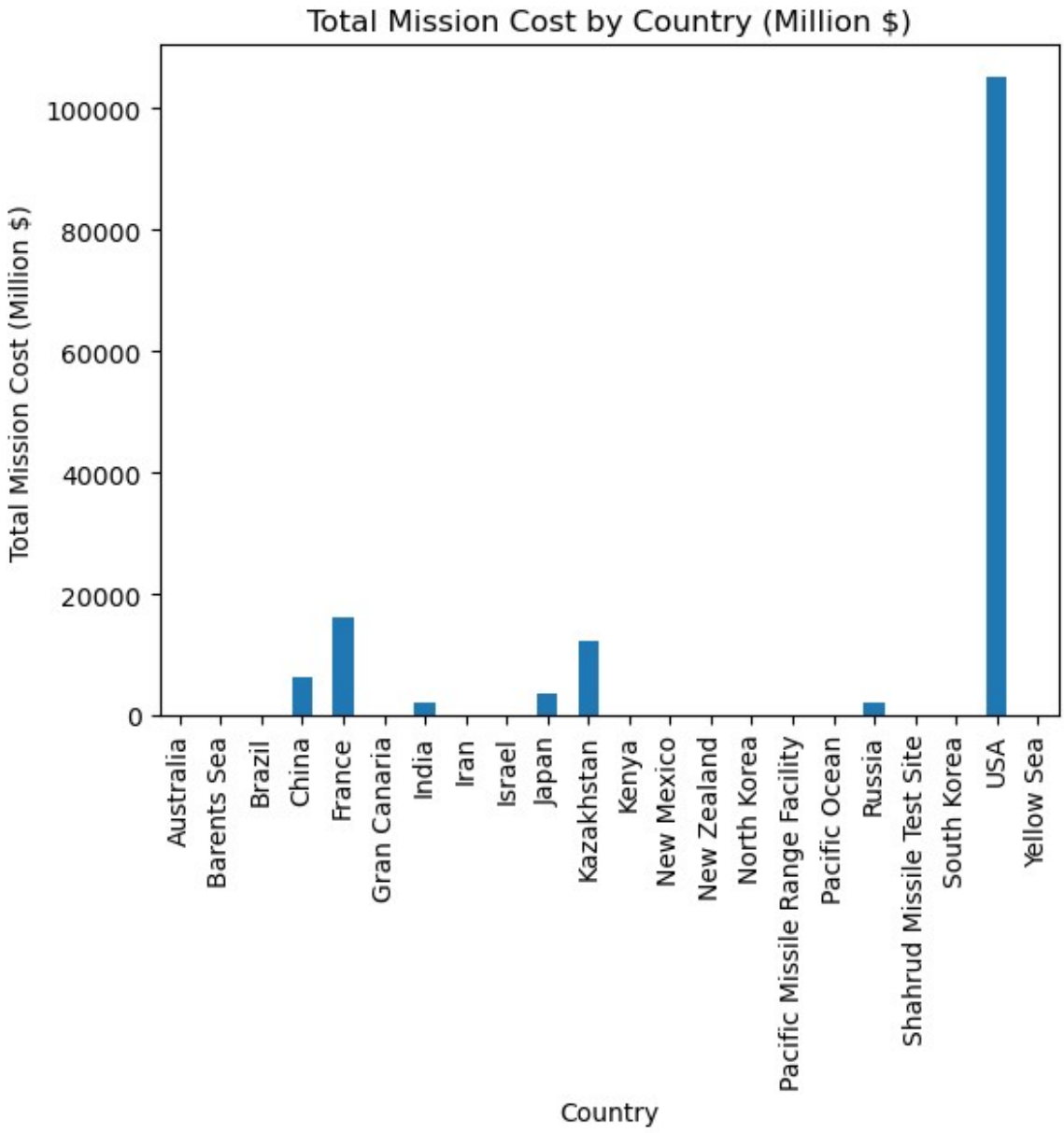
Country	SuccessFailureRatio
Australia	0.500000
Barents Sea	0.666667
Brazil	0.000000
China	0.906716
France	0.940594
Gran Canaria	1.000000
India	0.828947
Iran	0.307692
Israel	0.818182
Japan	0.896825
Kazakhstan	0.867332
Kenya	1.000000
New Mexico	0.000000

```
New Zealand                                0.846154
North Korea                                0.400000
Pacific Missile Range Facility             0.000000
Pacific Ocean                              0.916667
Russia                                     0.934145
Shahrud Missile Test Site                  1.000000
South Korea                                0.333333
USA                                        0.882440
Yellow Sea                                 1.000000
```

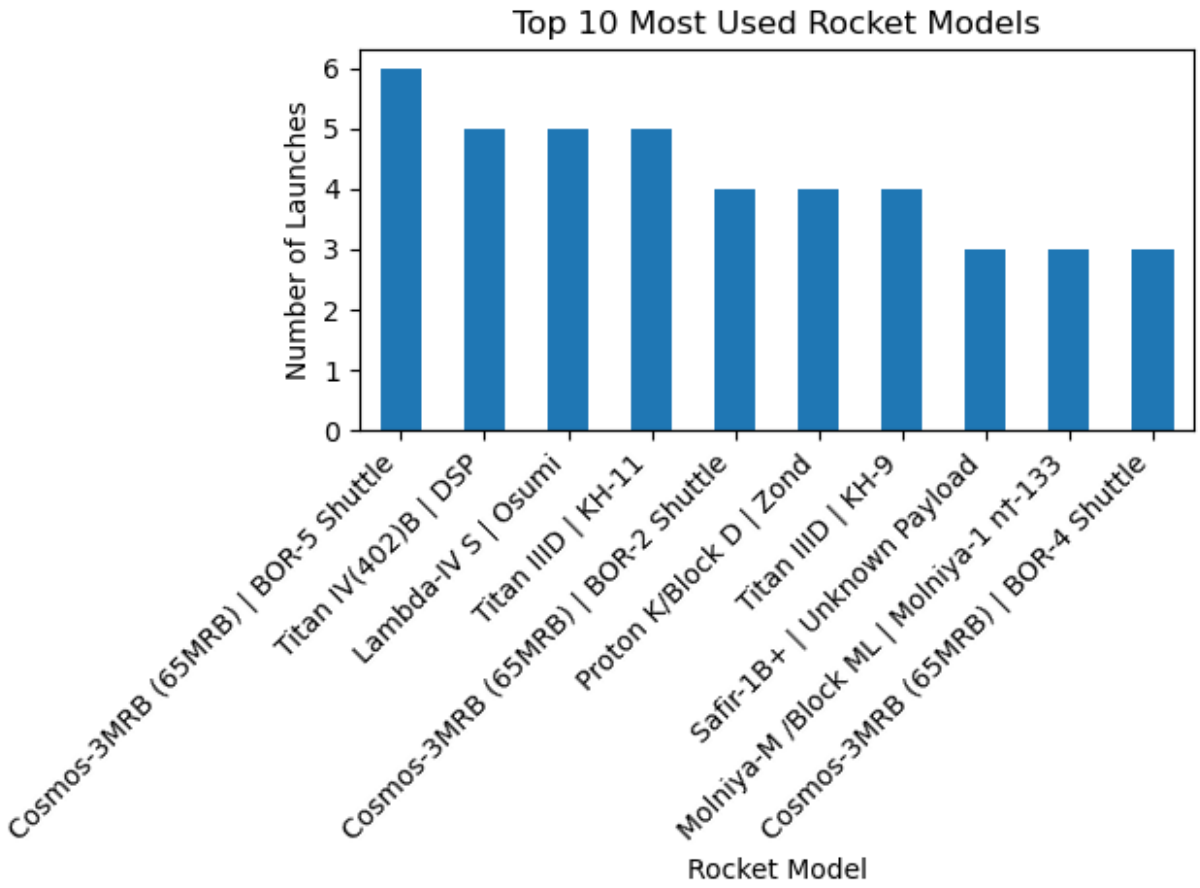
```
import matplotlib.pyplot as plt
missions_by_country = df['Country'].value_counts()
missions_by_country.plot(kind='bar', xlabel='Country', ylabel='Number
of Missions', title='Number of Missions by Country')
plt.show()
```



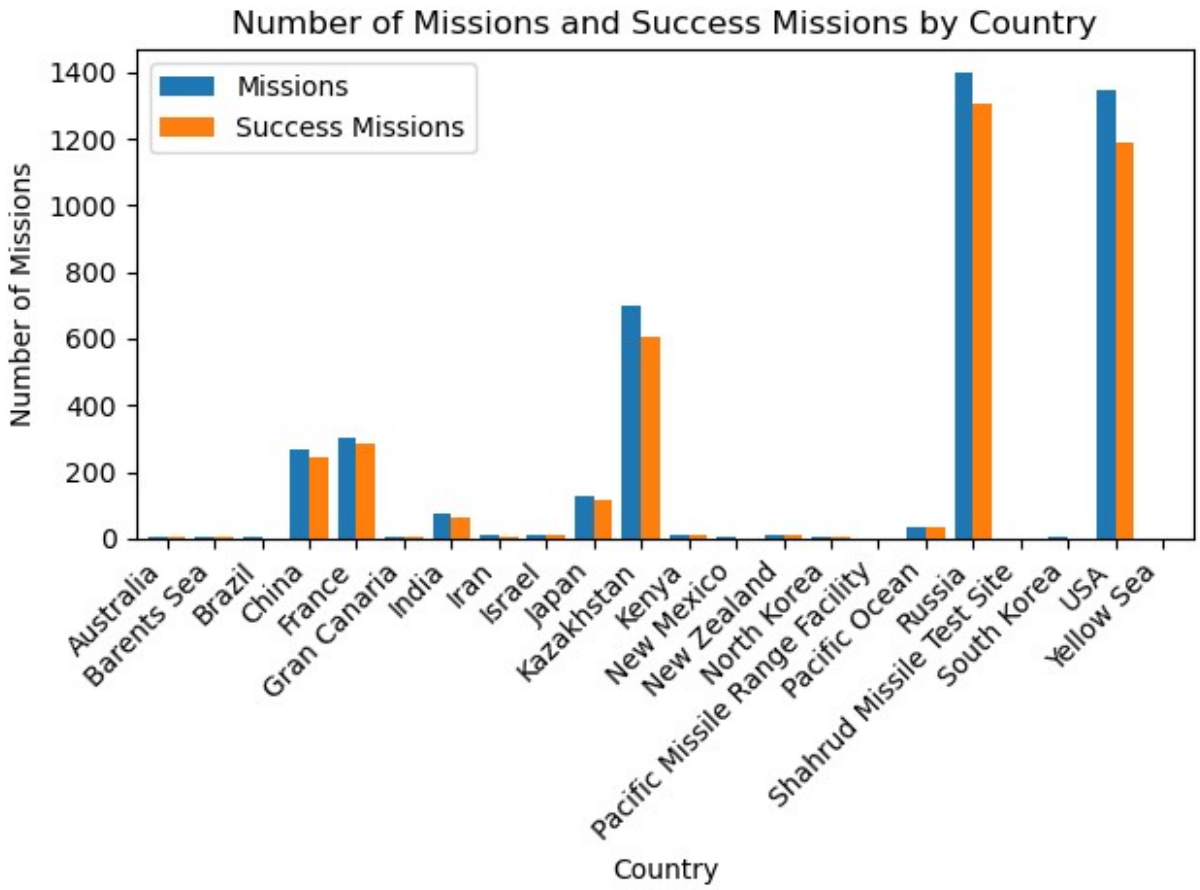

```
mission_costs_by_country = df.groupby('Country')['RocketCost'].sum()
mission_costs_by_country.plot(kind='bar', xlabel='Country',
ylabel='Total Mission Cost (Million $)', title='Total Mission Cost by
Country (Million $)')
plt.show()
```




```
rocket_models_count = df['RocketModel'].value_counts().head(10)


rocket_models_count.plot(kind='bar', xlabel='Rocket Model',
ylabel='Number of Launches', title='Top 10 Most Used Rocket Models')
plt.xticks(rotation=45, ha='right')  # Obrócenie etykiet osi x dla
lepszej czytelności
plt.tight_layout()  # Dostosowanie układu, aby zapobiec przecinaniu
się etykiet
plt.show()
```




```
missions_by_country = df['Country'].value_counts()
success_missions_by_country = df[df['MissionStatus'] == 'Success']
['Country'].value_counts()


missions_data = pd.DataFrame({'Missions': missions_by_country,
'Success Missions': success_missions_by_country})


missions_data.plot(kind='bar', xlabel='Country', ylabel='Number of
Missions', title='Number of Missions and Success Missions by Country',
width=0.8)
plt.xticks(rotation=45, ha='right')  # Obrócenie etykiet osi x dla
lepszej czytelności
plt.tight_layout()  # Dostosowanie układu, aby zapobiec przecinaniu
się etykiet
plt.show()
```




```
missions_by_country = df['Country'].value_counts()
success_missions_by_country = df[df['MissionStatus'] == 'Success']
['Country'].value_counts()


missions_data = pd.DataFrame({'Missions': missions_by_country,
'Success Missions': success_missions_by_country})


missions_data['Success Rate (%)'] = (missions_data['Success Missions']
/ missions_data['Missions']) * 100


print(missions_data)
```

```
                                  Missions  Success Missions  Success
Rate (%)
Country

Australia                                6               3.0
50.000000
Barents Sea                              3               2.0
```

```
66.666667
Brazil                                3                NaN
NaN
China                               268              243.0
90.671642
France                              303              285.0
94.059406
Gran Canaria                          2                2.0
100.000000
India                                76               63.0
82.894737
Iran                                 13                4.0
30.769231
Israel                               11                9.0
81.818182
Japan                               126              113.0
89.682540
Kazakhstan                          701              608.0
86.733238
Kenya                                 9                9.0
100.000000
New Mexico                            4                NaN
NaN
New Zealand                          13               11.0
84.615385
North Korea                           5                2.0
40.000000
Pacific Missile Range Facility        1                NaN
NaN
Pacific Ocean                        36               33.0
91.666667
Russia                             1397             1305.0
93.414460
Shahrud Missile Test Site             1                1.0
100.000000
South Korea                           3                1.0
33.333333
USA                                1344             1186.0
88.244048
Yellow Sea                            1                1.0
100.000000

average_mission_cost_by_country = df.groupby('Country')
['RocketCost'].mean()

print(average_mission_cost_by_country)

Country
Australia                                  NaN
Barents Sea                                NaN
```

```
Brazil                                    NaN
China                                40.273797
France                              171.421053
Gran Canaria                         40.000000
India                                32.492537
Iran                                       NaN
Israel                                     NaN
Japan                                92.512500
Kazakhstan                          264.141304
Kenya                                      NaN
New Mexico                                 NaN
New Zealand                           7.500000
North Korea                                NaN
Pacific Missile Range Facility       15.000000
Pacific Ocean                              NaN
Russia                               40.542593
Shahrud Missile Test Site                  NaN
South Korea                                NaN
USA                                 216.000657
Yellow Sea                            5.300000
Name: RocketCost, dtype: float64
```

```
AverageRocketCost=df['RocketCost'].mean()
print(AverageRocketCost)
```

```
153.79219917012446
```

```
average_rocket_cost_overall = df['RocketCost'].mean()

# Obliczenie średniego kosztu rakiety dla każdego kraju
average_rocket_cost_by_country = df.groupby('Country')
['RocketCost'].mean()

# Obliczenie procentowego porównania średniego kosztu rakiety danego
kraju do średniego kosztu rakiet ogólnie
percent_comparison = (average_rocket_cost_by_country /
average_rocket_cost_overall) * 100

# Utworzenie ramki danych zawierającej średni koszt rakiety dla
każdego kraju oraz procentowe porównanie do średniego kosztu rakiet
ogólnie
comparison_df = pd.DataFrame({'Average Rocket Cost by Country':
average_rocket_cost_by_country, 'AVG_rocketCostByCountry/AVGRocketCost
%': percent_comparison})

# Wyświetlenie tabeli z wynikami
print(comparison_df)
```

```
                                  Average Rocket Cost by Country  \
Country
```

Australia	NaN
Barents Sea	NaN
Brazil	NaN
China	40.273797
France	171.421053
Gran Canaria	40.000000
India	32.492537
Iran	NaN
Israel	NaN
Japan	92.512500
Kazakhstan	264.141304
Kenya	NaN
New Mexico	NaN
New Zealand	7.500000
North Korea	NaN
Pacific Missile Range Facility	15.000000
Pacific Ocean	NaN
Russia	40.542593
Shahrud Missile Test Site	NaN
South Korea	NaN
USA	216.000657
Yellow Sea	5.300000

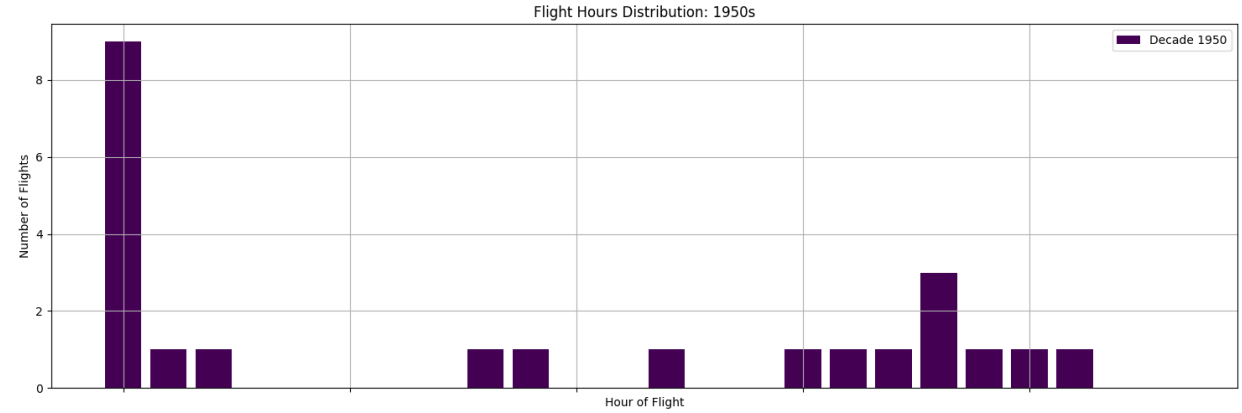
Country	AVG_rocketCostByCountry/AVGRocketCost%
Australia	NaN
Barents Sea	NaN
Brazil	NaN
China	26.187152
France	111.462775
Gran Canaria	26.009122
India	21.127559
Iran	NaN
Israel	NaN
Japan	60.154221
Kazakhstan	171.752082
Kenya	NaN

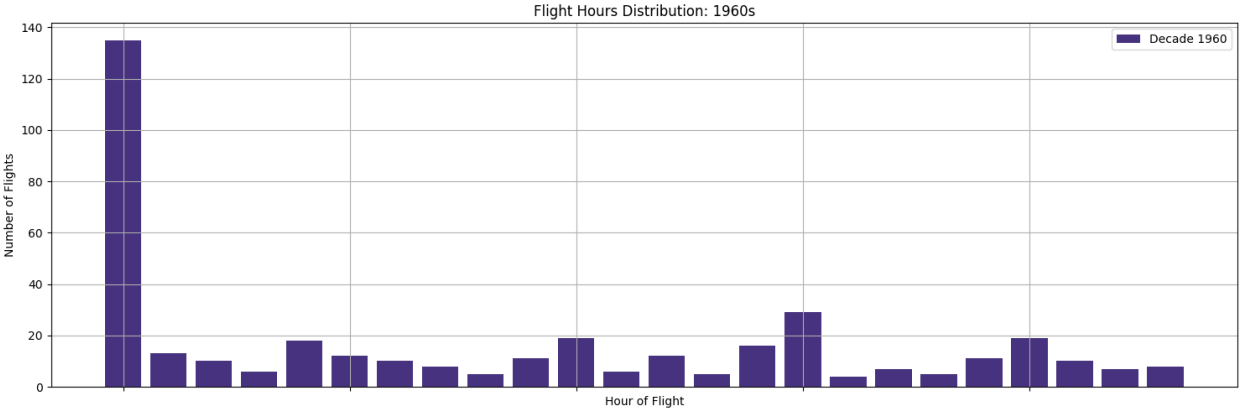
```
New Mexico                                                    NaN
New Zealand                                                 4.876710
North Korea                                                   NaN
Pacific Missile Range Facility                              9.753421
Pacific Ocean                                                 NaN
Russia                                                     26.361930
Shahrud Missile Test Site                                     NaN
South Korea                                                   NaN
USA                                                       140.449684
Yellow Sea                                                  3.446209
```

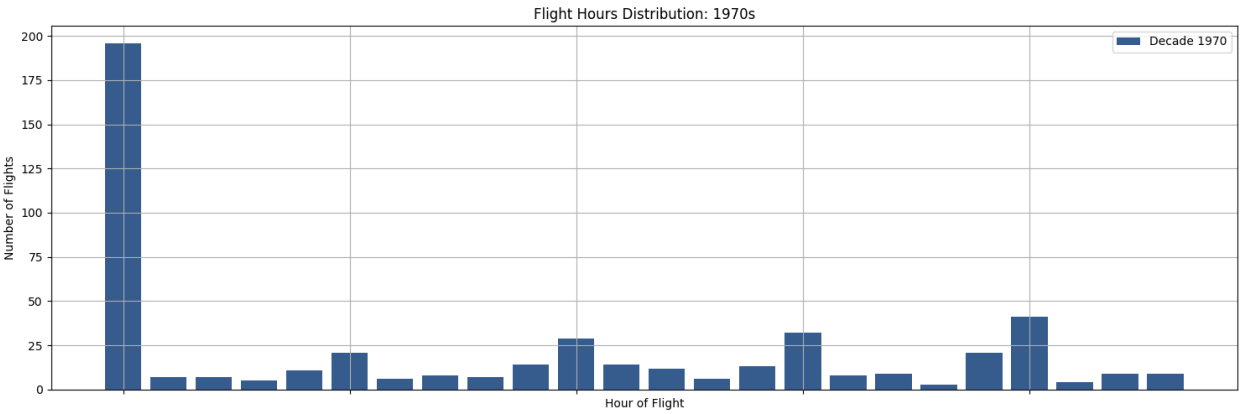
```
model_count = df['RocketModel'].value_counts()

top_20_models = model_count.head(20)

print(top_20_models)
```

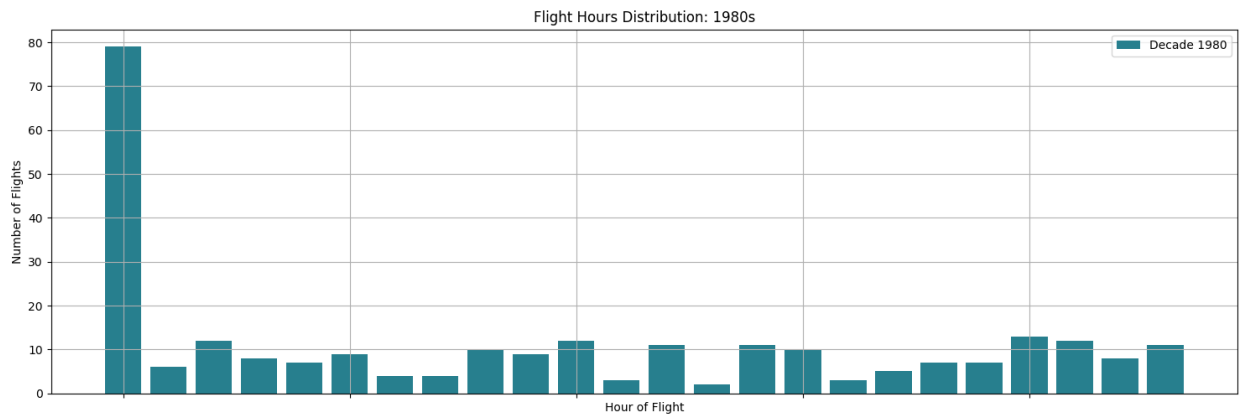
```
RocketModel
Cosmos-3MRB (65MRB) | BOR-5 Shuttle           6
Titan IV(402)B | DSP                          5
Lambda-IV S | Osumi                           5
Titan IIID | KH-11                            5
Cosmos-3MRB (65MRB) | BOR-2 Shuttle           4
Proton K/Block D | Zond                       4
Titan IIID | KH-9                             4
Safir-1B+ | Unknown Payload                   3
Molniya-M /Block ML | Molniya-1 n†133         3
Cosmos-3MRB (65MRB) | BOR-4 Shuttle           3
Titan IV(402)A | DSP                          3
Cosmos-3M (11K65M) | Strela-2M satellite      3
Cosmos-3MRB (65MRB) | BOR-3 Shuttle           2
Voskhod | Cosmos 554                         2
Titan IV(401)A | Mercury                      2
Long March 2D | Shiyan-3 & Chuangxin-1(02)    2
Cosmos-3M (11K65M) | Cosmos 2265              2
Titan IV(401)A | Trumpet                      2
Cosmos-3 (11K65) | VKZ                        2
```

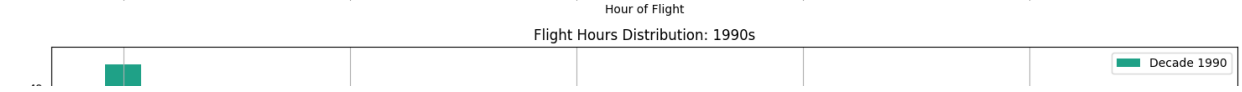
```
Name: count, dtype: int64

import matplotlib.pyplot as plt
import pandas as pd
import numpy as np



df['Date'] = pd.to_datetime(df['Date'], errors='coerce')
df['Decade'] = (df['Date'].dt.year // 10) * 10

df['Hour'] = pd.to_datetime(df['Time'], format='%H:%M.%S',
errors='coerce').dt.hour


df_cleaned = df.dropna(subset=['Decade', 'Hour'])


grouped_data_decade = df_cleaned.groupby(['Decade',
'Hour']).size().reset_index(name='Flights')


decades = sorted(df_cleaned['Decade'].unique())
colors = plt.cm.viridis(np.linspace(0, 1, len(decades)))

fig, axs = plt.subplots(len(decades), 1, figsize=(15, 5 *
len(decades)), sharex=True)

for i, decade in enumerate(decades):
    ax = axs[i]
    decade_data = grouped_data_decade[grouped_data_decade['Decade'] ==
decade]
    ax.bar(decade_data['Hour'], decade_data['Flights'],
color=colors[i], label=f'Decade {int(decade)}')
    ax.set_title(f'Flight Hours Distribution: {int(decade)}s')
    ax.set_xlabel('Hour of Flight')
    ax.set_ylabel('Number of Flights')
    ax.legend()
    ax.grid(True)

plt.tight_layout()
plt.show()
```


Flight Hours Distribution: 1950s
Decade 1950
Number of Flights
Hour of Flight


Flight Hours Distribution: 1960s
Decade 1960
Number of Flights
Hour of Flight


Flight Hours Distribution: 1970s
Decade 1970
Number of Flights
Hour of Flight

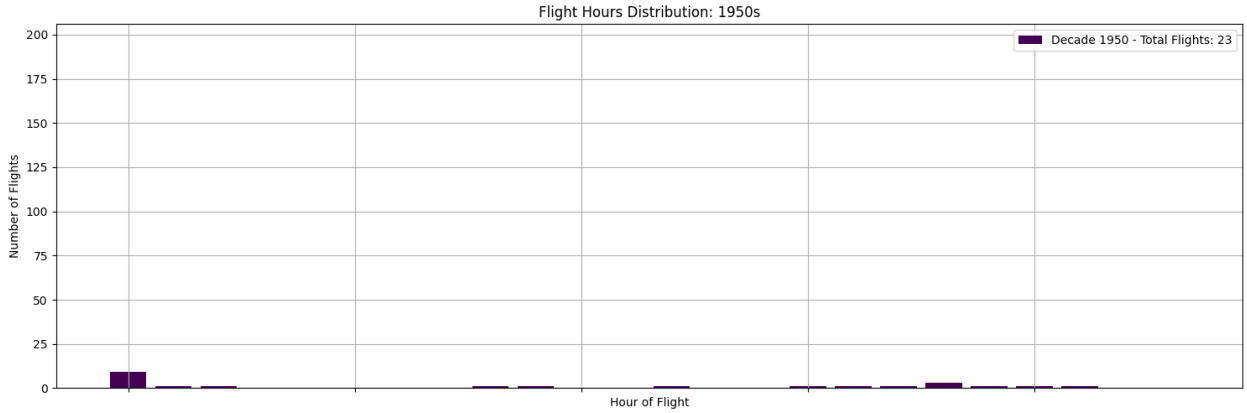

Flight Hours Distribution: 1980s
Decade 1980
Number of Flights
Hour of Flight

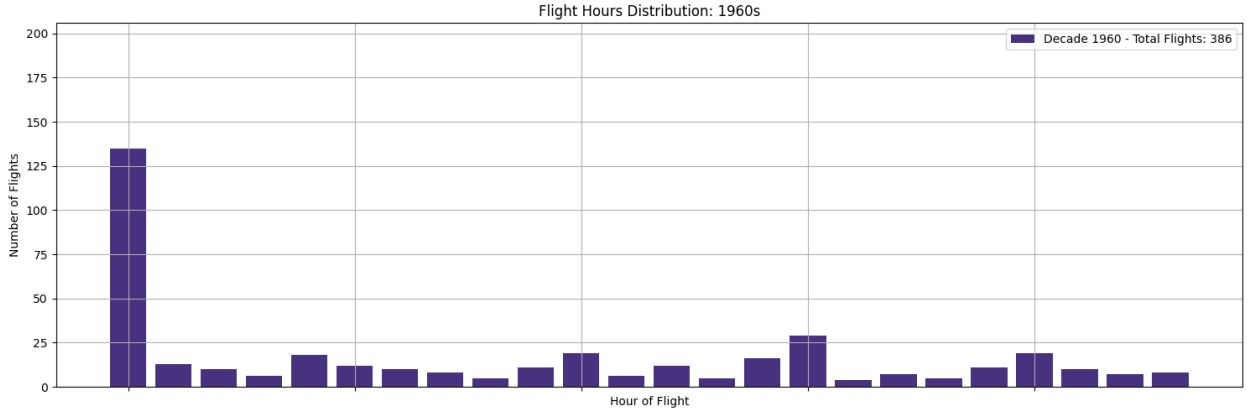

Flight Hours Distribution: 1990s
Decade 1990

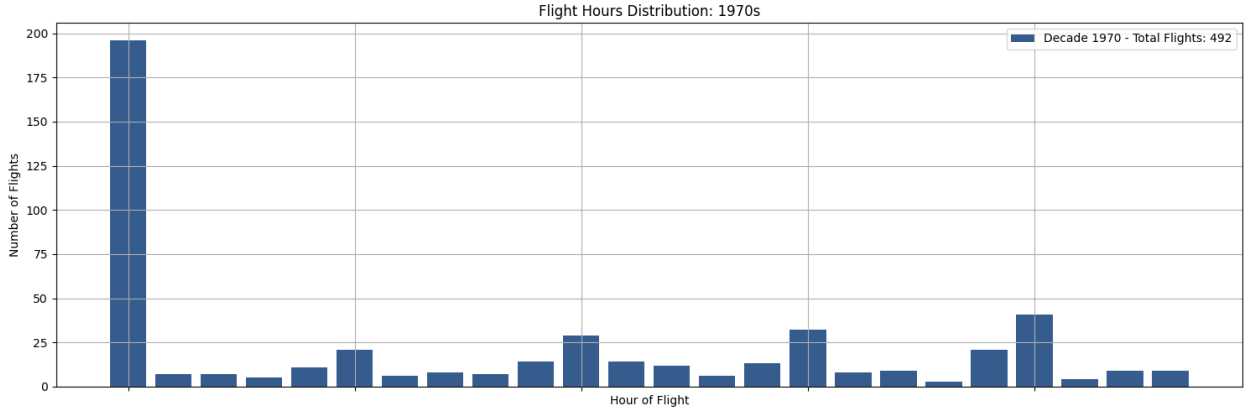
```
# Wyszukanie pierwszej misji SpaceX w dostarczonych danych
first_spacex_mission = df[df['CompanyName'].str.contains('SpaceX',
case=False)].sort_values('Date').iloc[0]

first_spacex_mission_date = first_spacex_mission['Date']
first_spacex_mission_date
```

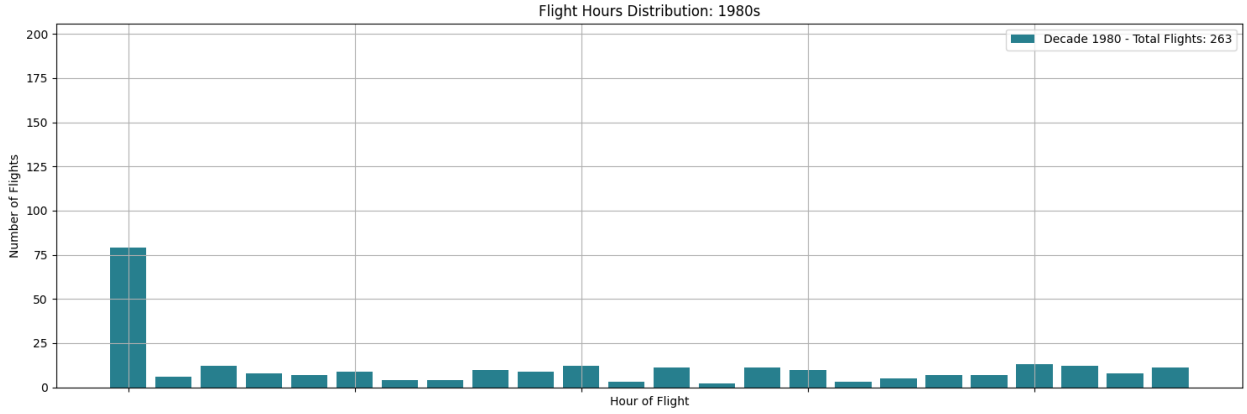
```
Timestamp('2006-03-24 00:00:00')
```

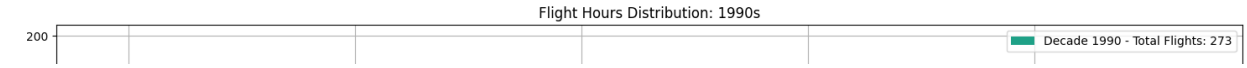
```
# Ponowne tworzenie wykresów z tą samą skalą dla wszystkich dekad i
dodanie liczby misji dla każdej dekady

# Obliczenie maksymalnej liczby lotów w jednej godzinie przez
wszystkie dekady, aby ustawić tę samą skalę na osi Y
max_flights = grouped_data_decade['Flights'].max()

# Tworzenie wykresów
fig, axs = plt.subplots(len(decades), 1, figsize=(15, 5 *
len(decades)), sharex=True, sharey=True)

for i, decade in enumerate(decades):
    ax = axs[i]
    decade_data = grouped_data_decade[grouped_data_decade['Decade'] ==
decade]
    total_flights_decade = decade_data['Flights'].sum()  # Suma lotów
dla dekady
    ax.bar(decade_data['Hour'], decade_data['Flights'],
color=colors[i], label=f'Decade {int(decade)} - Total Flights:
{total_flights_decade}')
    ax.set_title(f'Flight Hours Distribution: {int(decade)}s')
    ax.set_xlabel('Hour of Flight')
    ax.set_ylabel('Number of Flights')
    ax.set_ylim(0, max_flights + 10)
    ax.legend()
    ax.grid(True)

plt.tight_layout()
plt.show()
```


Flight Hours Distribution: 1950s
Decade 1950 - Total Flights: 23
Number of Flights
Hour of Flight


Flight Hours Distribution: 1960s
Decade 1960 - Total Flights: 386
Number of Flights
Hour of Flight


Flight Hours Distribution: 1970s
Decade 1970 - Total Flights: 492
Number of Flights
Hour of Flight

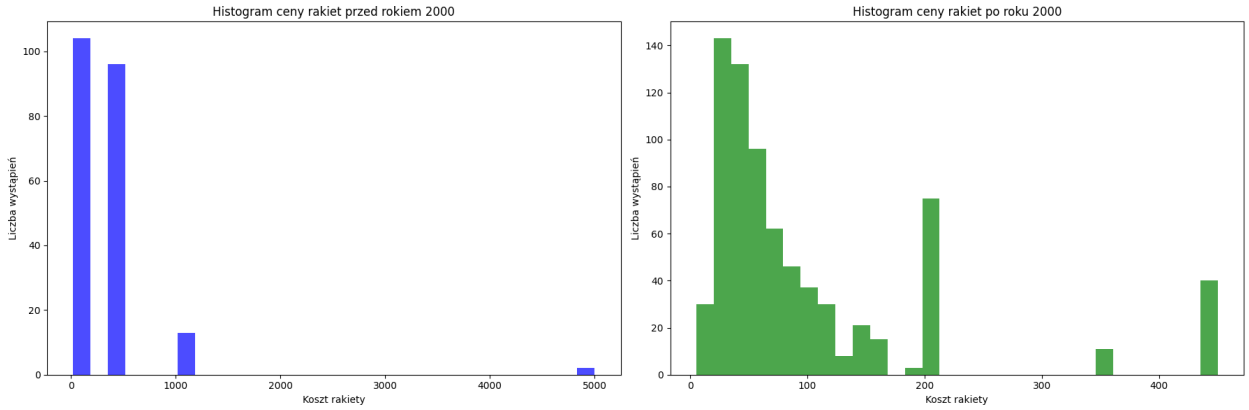

Flight Hours Distribution: 1980s
Decade 1980 - Total Flights: 263
Number of Flights
Hour of Flight


Flight Hours Distribution: 1990s
Decade 1990 - Total Flights: 273

```
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')


filtered_data = df.dropna(subset=['RocketCost'])


before_2000 = filtered_data[filtered_data['Date'].dt.year < 2000]
after_2000 = filtered_data[filtered_data['Date'].dt.year >= 2000]


average_cost_before_2000 = before_2000['RocketCost'].mean()
average_cost_after_2000 = after_2000['RocketCost'].mean()

average_cost_before_2000, average_cost_after_2000
```

```
(340.2252558139535, 100.27670226969292)
```

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

fig, ax = plt.subplots(1, 2, figsize=(18, 6))

ax[0].hist(before_2000['RocketCost'], bins=30, alpha=0.7,
color='blue')
ax[0].set_title('Histogram ceny rakiet przed rokiem 2000')
ax[0].set_xlabel('Koszt rakiety')
ax[0].set_ylabel('Liczba wystąpień')


ax[1].hist(after_2000['RocketCost'], bins=30, alpha=0.7,
color='green')
ax[1].set_title('Histogram ceny rakiet po roku 2000')
ax[1].set_xlabel('Koszt rakiety')
ax[1].set_ylabel('Liczba wystąpień')


plt.tight_layout()
plt.show()
```




```
before_2000['RocketCost'].count()
```

```
215
```

```
after_2000['RocketCost'].count()
```

```
749
```

```
# Obliczenie maksymalnego kosztu rakiety przed rokiem 2000
max_cost_before_2000 = before_2000['RocketCost'].max()

# Obliczenie liczby wartości równych maksymalnemu kosztowi przed
rokiem 2000
max_cost_before_2000_count = before_2000[before_2000['RocketCost'] ==
max_cost_before_2000].shape[0]

max_cost_before_2000_count
```

```
2
```

```
# Obliczenie liczby wystąpień poszczególnych wartości kosztów rakiet
przed rokiem 2000
cost_counts_before_2000 =
before_2000['RocketCost'].value_counts().reset_index()
cost_counts_before_2000.columns = ['RocketCost', 'Count']

cost_counts_before_2000
```

	RocketCost	Count
0	450.00	96
1	40.00	28
2	59.00	22
3	30.80	18
4	1160.00	13
5	35.00	9
6	25.00	5
7	136.60	4

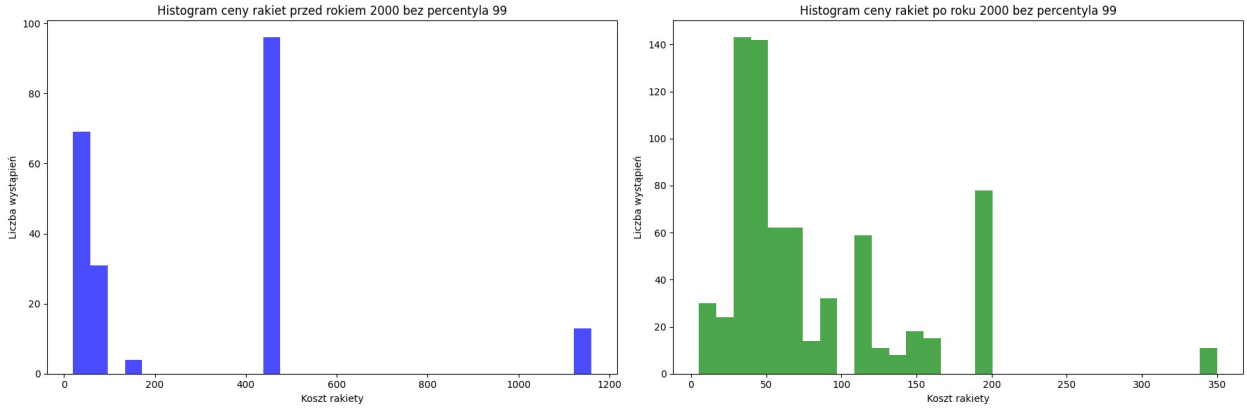
```
8         63.23      4
9         45.00      4
10        69.70      3
11        29.75      3
12      5000.00      2
13        64.68      2
14        20.00      1
15        29.00      1
```

```
# Obliczenie 99. percentyla dla cen rakiet
percentile_99_before_2000 = before_2000['RocketCost'].quantile(1)
percentile_99_after_2000 = after_2000['RocketCost'].quantile(1)

# Filtrowanie danych, aby usunąć wartości powyżej 99. percentyla
filtered_before_2000 = before_2000[before_2000['RocketCost'] <
percentile_99_before_2000]
filtered_after_2000 = after_2000[after_2000['RocketCost'] <
percentile_99_after_2000]

# Wyświetlenie liczby wierszy przed i po usunięciu outlierów
len_before_2000_removed = before_2000.shape[0] -
filtered_before_2000.shape[0]
len_after_2000_removed = after_2000.shape[0] -
filtered_after_2000.shape[0]

len_before_2000_removed, len_after_2000_removed
```

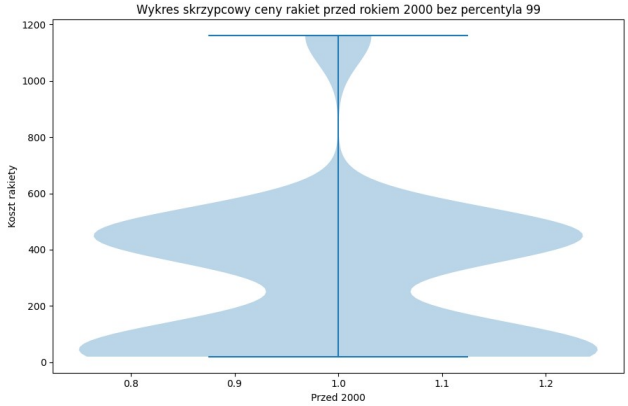
```
(2, 40)
```

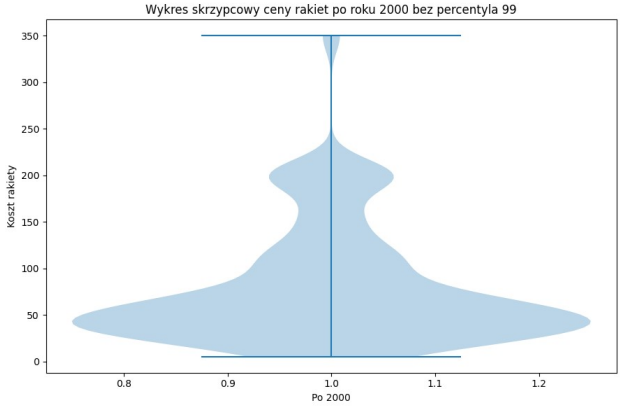
```
# Ponowne ustawienie wykresów po usunięciu outlierów
fig, ax = plt.subplots(1, 2, figsize=(18, 6))

# Histogram dla cen rakiet przed rokiem 2000 bez percentyla 99
ax[0].hist(filtered_before_2000['RocketCost'], bins=30, alpha=0.7,
color='blue')
ax[0].set_title('Histogram ceny rakiet przed rokiem 2000 bez
percentyla 99')
ax[0].set_xlabel('Koszt rakiety')
ax[0].set_ylabel('Liczba wystąpień')

# Histogram dla cen rakiet po roku 2000 bez percentyla 99
# (Brak zmian, ponieważ nie usunięto żadnych wartości)
ax[1].hist(filtered_after_2000['RocketCost'], bins=30, alpha=0.7,
color='green')
ax[1].set_title('Histogram ceny rakiet po roku 2000 bez percentyla
99')
ax[1].set_xlabel('Koszt rakiety')
ax[1].set_ylabel('Liczba wystąpień')

# Wyświetlenie wykresów
```

```
plt.tight_layout()
plt.show()
```




```
# Ponowne ustawienie wykresów jako wykresy skrzypcowe (violin plots) po usunięciu outlierów
fig, ax = plt.subplots(1, 2, figsize=(18, 6))

# Wykres skrzypcowy dla cen rakiet przed rokiem 2000 bez percentyla 99
ax[0].violinplot(filtered_before_2000['RocketCost'])
ax[0].set_title('Wykres skrzypcowy ceny rakiet przed rokiem 2000 bez percentyla 99')
ax[0].set_xlabel('Przed 2000')
ax[0].set_ylabel('Koszt rakiety')

# Wykres skrzypcowy dla cen rakiet po roku 2000 bez percentyla 99
# (Brak zmian, ponieważ nie usunięto żadnych wartości)
ax[1].violinplot(filtered_after_2000_no_max['RocketCost'])
ax[1].set_title('Wykres skrzypcowy ceny rakiet po roku 2000 bez percentyla 99')
ax[1].set_xlabel('Po 2000')
ax[1].set_ylabel('Koszt rakiety')

# Wyświetlenie wykresów
plt.tight_layout()
plt.show()
```


Wykres skrzypcowy ceny rakiet przed rokiem 2000 bez percentyla 99
Koszt rakiety
Przed 2000
0
200
400
600
800
1000
1200
0.8
0.9
1.0
1.1
1.2


Wykres skrzypcowy ceny rakiet po roku 2000 bez percentyla 99
Koszt rakiety
Po 2000
0
50
100
150
200
250
300
350
0.8
0.9
1.0
1.1
1.2