Space Race Analysis and Visualization

The Space Race project studies more than 4300 space missions from 1957 to 2020. We look at who's involved, when missions happened, what they aimed to do, how successful they were, how much they cost, and if the rockets used are still in use. We want to see patterns in how many launches there are, how history like the Cold War affected space activities, and which countries and groups are most active in space exploration. Our goal is to understand how space exploration has changed over time and why.

Data Dictionary:

Variable	Description
Unnamed: 0.1	Junk column
Unnamed: 0	Junk column
Organisation	Organisation, that created and launched the rocket
Location	Location from where the rocket was launched
Date	Mission Date
Detail	Rocket Name
Rocket_Status	Represents if rocket is still active or retired
Price	Price of the whole mission (millions)
Mission_Status	Shows whether the mission was successful or not

In [7]:
```
df.head(20)
```

Out[7]:

	Unnamed: 0.1	Unnamed: 0	Organisation	Location	Date	Detail	Rocket_Status	Price	Mission_Status
0	0	0	SpaceX	LC-39A, Kennedy Space Center, Florida, USA	Fri Aug 07, 2020 05:12 UTC	Falcon 9 Block 5 | Starlink V1 L9 & BlackSky	StatusActive	50.0	Success
1	1	1	CASC	Site 9401 (SLS-2), Jiuquan Satellite Launch Ce...	Thu Aug 06, 2020 04:01 UTC	Long March 2D | Gaofen-9 04 & Q-SAT	StatusActive	29.75	Success
2	2	2	SpaceX	Pad A, Boca Chica, Texas, USA	Tue Aug 04, 2020 23:57 UTC	Starship Prototype | 150 Meter Hop	StatusActive	NaN	Success
3	3	3	Roscosmos	Site 200/39, Baikonur Cosmodrome, Kazakhstan	Thu Jul 30, 2020 21:25 UTC	Proton-M/Briz-M | Ekspress-80 & Ekspress-103	StatusActive	65.0	Success
4	4	4	ULA	SLC-41, Cape Canaveral AFS, Florida, USA	Thu Jul 30, 2020 11:50 UTC	Atlas V 541 | Perseverance	StatusActive	145.0	Success
5	5	5	CASC	LC-9, Taiyuan Satellite Launch Center, China	Sat Jul 25, 2020 03:13 UTC	Long March 4B | Ziyuan-3 03, Apocalypse-10 & N...	StatusActive	64.68	Success

Data Preprocessing

In [111]:
```
df.shape
```

Out[111]:
```
(4324, 9)
```

In [112]:
```
df.columns
```

Out[112]:
```
Index(['Unnamed: 0.1', 'Unnamed: 0', 'Organisation', 'Location', 'Date',
       'Detail', 'Rocket_Status', 'Price', 'Mission_Status'],
      dtype='object')
```

In [113]:
```
df['Price'] = df['Price'].apply(lambda x:str(x).replace(',','')).astype('float64')
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')
print(df.dtypes)
```

```
Unnamed: 0.1                int64
Unnamed: 0                  int64
Organisation               object
Location                   object
Date             datetime64[ns, UTC]
Detail                     object
Rocket_Status              object
Price                     float64
Mission_Status             object
dtype: object
```

In [116]:
```
df.isna().sum()
```

Out[116]:
```
Unnamed: 0.1        0
Unnamed: 0          0
Organisation        0
Location            0
Date              126
Detail              0
Rocket_Status       0
Price            3360
Mission_Status      0
dtype: int64
```

Data Cleaning

In [13]:
```
#df['Price'] = df['Price'].fillna(df['Price'].mean().round(2))
#df.dropna(subset=['Price'], inplace=True)
df['Price'].fillna(0, inplace = True)
df['Date'].fillna(method='ffill', inplace=True)
print(df.isna().sum())
```

```
Unnamed: 0.1      0
Unnamed: 0        0
Organisation      0
Location          0
Date              0
Detail            0
Rocket_Status     0
Price             0
Mission_Status    0
dtype: int64
```

In [16]:
```
df = df.drop(["Unnamed: 0.1", "Unnamed: 0"], axis=1)
```

In [17]:
```
symbols = "@$%^*=*?\<>`~Â"

for column in df.columns:
    if df[column].dtype != 'object':
        continue
    symbols_found = df[column].apply(lambda x: any(char in symbols for char in x))
    rows_with_symbols = df[symbols_found]
    if not rows_with_symbols.empty:
        print(f"Symbols found in '{column}' column:")
        print(rows_with_symbols)
```

```
Symbols found in 'Organisation' column:
        Organisation                                Location
3800  Arm??e de l'Air  Brigitte, Hammaguir, Algeria, France  \
3803  Arm??e de l'Air  Brigitte, Hammaguir, Algeria, France
3903  Arm??e de l'Air  Brigitte, Hammaguir, Algeria, France
3923  Arm??e de l'Air  Brigitte, Hammaguir, Algeria, France

                          Date                  Detail  Rocket_Status  Price
3800 1967-02-15 10:06:00+00:00  Diamant A | Diad?ªme 2  StatusRetired   0.00  \
3803 1967-02-08 08:39:00+00:00  Diamant A | Diad?ªme 1  StatusRetired   0.00
3903 1966-02-17 07:33:00+00:00    Diamant A | Diapason  StatusRetired   0.00
3923 1965-11-26 14:47:00+00:00    Diamant A | Ast??rix  StatusRetired   0.00

      Mission_Status
3800         Success
3803  Partial Failure
3903         Success
3923         Success
Symbols found in 'Location' column:
     Organisation                                                Location
15     Rocket Lab    Rocket Lab LC-1A, M?□hia Peninsula, New Zealand  \
21     Rocket Lab    Rocket Lab LC-1A, M?□hia Peninsula, New Zealand
55     Rocket Lab    Rocket Lab LC-1A, M?□hia Peninsula, New Zealand
77     Rocket Lab    Rocket Lab LC-1A, M?□hia Peninsula, New Zealand
93     Rocket Lab    Rocket Lab LC-1A, M?□hia Peninsula, New Zealand
```

	Detail	Rocket_Status	Price	\
15	Electron/Curie | Pics Or It Didn??¦t Happen	StatusActive	7.50	\
60	Long March 2D | Jilin-1 Wideband 01 & ??uSat-7/8	StatusActive	29.75	
64	Rokot/Briz KM | Gonets-M ???24, 25, 26 [block-...	StatusRetired	41.80	
391	Vega | G??kt??rk-1A	StatusActive	37.00	
436	Long March 4B | Ziyuan III-02 & ??uSat-1, 2	StatusActive	64.68	
504	Falcon 9 v1.1 | Turkmen??lem52E/MonacoSat	StatusRetired	56.50	
546	Ariane 5 ES | Georges Lema??tre ATV	StatusRetired	0.00	
626	Soyuz ST-A/Fregat | Pl??iades 1B	StatusActive	80.00	
660	Soyuz ST-A/Fregat | Pl??iades 1A, SSOT, Elisa	StatusActive	80.00	
1224	Ariane 40 | Helios 1B & Cl??mentine	StatusRetired	0.00	
1266	Delta II 7920-10 | ARGOS (P91-1 ARGOS), ?örste...	StatusRetired	0.00	
1360	Titan IV(401)B | Cassini???Huygens	StatusRetired	0.00	
1435	Ariane 44L | Arabsat-2A, T??rksat 1C	StatusRetired	0.00	
1746	Cosmos-3M (11K65M) | Ta??foun n†59	StatusRetired	0.00	
2290	Cosmos-3M (11K65M) | Ta??foun n†32	StatusRetired	0.00	
2351	Cosmos-3M (11K65M) | Ta??foun n†27	StatusRetired	0.00	
2911	Saturn IB | ASTP (Apollo???Soyuz Test Project)	StatusRetired	0.00	
3399	Diamant B | P??ole	StatusRetired	0.00	
3570	Proton K/Block D | M-69 ???522	StatusRetired	0.00	
3800	Diamant A | Diad?ªme 2	StatusRetired	0.00	
3803	Diamant A | Diad?ªme 1	StatusRetired	0.00	
3923	Diamant A | Ast??rix	StatusRetired	0.00	

Noticing that there are many values that have char and other symbols in the middle of word letters, I am trying to find all the lines where any column has unclear or mismatched symbols.

In [18]:
```
# Char valymas
symbols = "!@#$%^*=*?\.<>`¦~Â"
df['Organisation'] = df['Organisation'].apply(lambda x: ''.join(char for char in x if char not in symbols))
df['Detail'] = df['Detail'].apply(lambda x: ''.join(char for char in x if char not in symbols))
df['Location'] = df['Location'].apply(lambda x: ''.join(char for char in x if char not in symbols))
```

In [19]:
```
def clean_column(df, column_name):
    final_str_column = []
    for detail in df[column_name]:
        # Use regular expression to remove non-printable characters
        clean_detail = re.sub(r'[^\x20-\x7E]', '', detail)
        final_str_column.append(clean_detail)
    df[column_name] = final_str_column

clean_column(df, 'Organisation')
clean_column(df, 'Detail')
clean_column(df, 'Location')

df.iloc[15]
```

Out[19]:
```
Organisation                                  Rocket Lab
Location          Rocket Lab LC-1A, Mhia Peninsula, New Zealand
Date                                2020-07-04 21:19:00+00:00
Detail               Electron/Curie | Pics Or It Didnt Happen
Rocket_Status                                   StatusActive
Price                                               7.50
Mission_Status                                      Failure
Name: 15, dtype: object
```

Then I create an algorithm that cleans any char or other non-printable characters in the middle of word letters. Also I use regular expression to correct this problem.

```
In [20]: duplicated_rows = df[df.duplicated(keep=False)]
         print(duplicated_rows)
         df = df.drop_duplicates()
```

```
    Organisation                                          Location
792         CASC  Site 9401 (SLS-2), Jiuquan Satellite Launch Ce...  \
793         CASC  Site 9401 (SLS-2), Jiuquan Satellite Launch Ce...

                          Date                                    Detail
792 2008-11-05 00:15:00+00:00  Long March 2D | Shiyan-3 & Chuangxin-1(02)  \
793 2008-11-05 00:15:00+00:00  Long March 2D | Shiyan-3 & Chuangxin-1(02)

    Rocket_Status  Price Mission_Status
792  StatusActive  29.75        Success
793  StatusActive  29.75        Success
```

The dataset is now clean. We dealt with the missing values in the Price column in three ways: 1. Filling them with the average price 2. Removing rows with missing prices 3. Setting missing prices to 0. However, since about 78% of the prices were missing (3375 out of 4323), using the average or 0 wouldn't provide meaningful insights. Removing rows would lose a lot of data, so we only did it for certain categorical variables when calculating statistics. We also removed any duplicate entries from the dataset to improve its quality.

Descriptive statistics

```
In [125]: df.describe()
```

Out[125]:

	Price
count	963.00
mean	153.92
std	288.57
min	5.30
25%	40.00
50%	62.00
75%	164.00
max	5,000.00

In [96]:
```
# Central tendency
print("Number of organisations: " + str(df["Organisation"].nunique()))

print("Earliest Date: " + str(df['Date'].min()))
print("Latest Date: " + str(df['Date'].max()))
print('-----------------------------------------------')
print("Number of rocket details: " + str(df["Detail"].nunique()))
df["Price"] = pd.to_numeric(df["Price"], errors='coerce')
print("The average price of rocket launch: " + str(round(df["Price"].mean(), 2)))
print("The middle price of rocket launch: " + str(df["Price"].median()))
print("The most frequently occurring price of rocket launch: " + str(df["Price"].mode().tolist()))
print('-----------------------------------------------')


# Range
Range = df["Price"].max() - df["Price"].min()
print("A range of prices: " + str(Range))
print('-----------------------------------------------')

# Variance
price_variance = df["Price"].var()
rounded_variance = round(price_variance, 2)
print("A measure of how spread out the prices of rocket launches are: " + str(rounded_variance))
print('-----------------------------------------------')

# Standard deviation
s_dev = math.sqrt(rounded_variance)
print("The square root of the variance of rocket launch prices: " + str(round(s_dev, 2)))
```

```
Number of organisations: 24
Earliest Date: 1964-09-01 15:00:00+00:00
Latest Date: 2020-08-07 05:12:00+00:00
-----------------------------------------------
Number of rocket details: 947
The average price of rocket launch: 129.9
The middle price of rocket launch: 62.0
The most frequently occurring price of rocket launch: [450.0]
-----------------------------------------------
A range of prices: 444.7
-----------------------------------------------
A measure of how spread out the prices of rocket launches are: 20523.14
-----------------------------------------------
The square root of the variance of rocket launch prices: 143.26
```

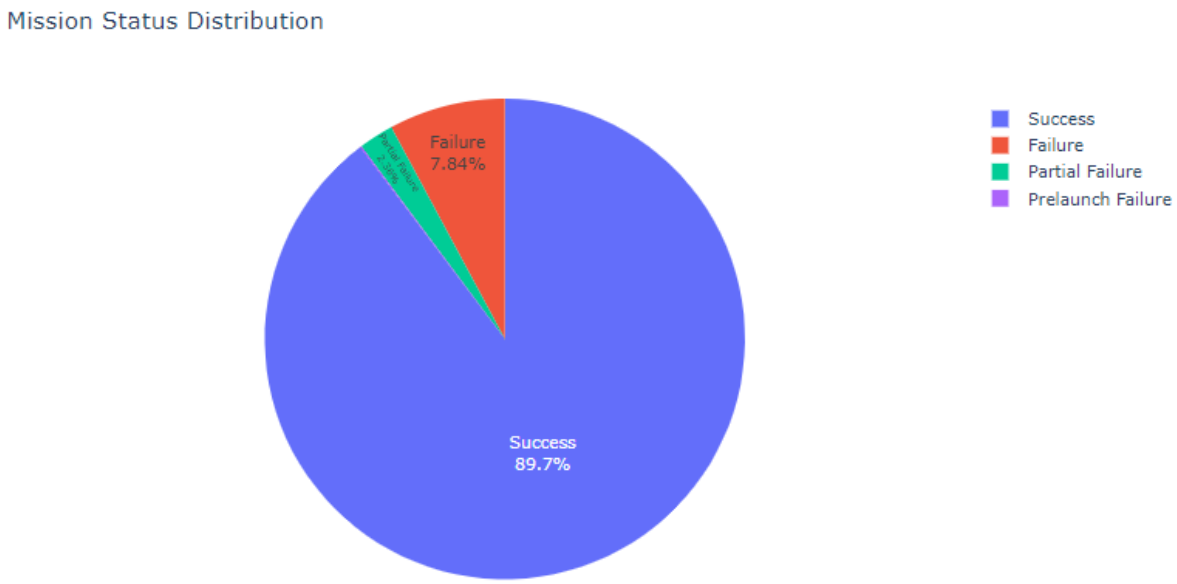
The outlier in the dataset is the maximum price of 5000, which represents the investment in developing Proton rockets for the Russian space program. However, it's important to note that the Energiya/Buran program, which also cost around 5 billion rubles, was separate from the Proton rocket program. Energiya/Buran involved the development of the Energia rocket and the Buran spacecraft. So, while the Proton rockets were a significant investment for Russia, the cost of Energiya/Buran was not directly related to the Proton rocket program.

Number of Launches per Company
Number of Active versus Retired Rockets

Distribution of Mission Status

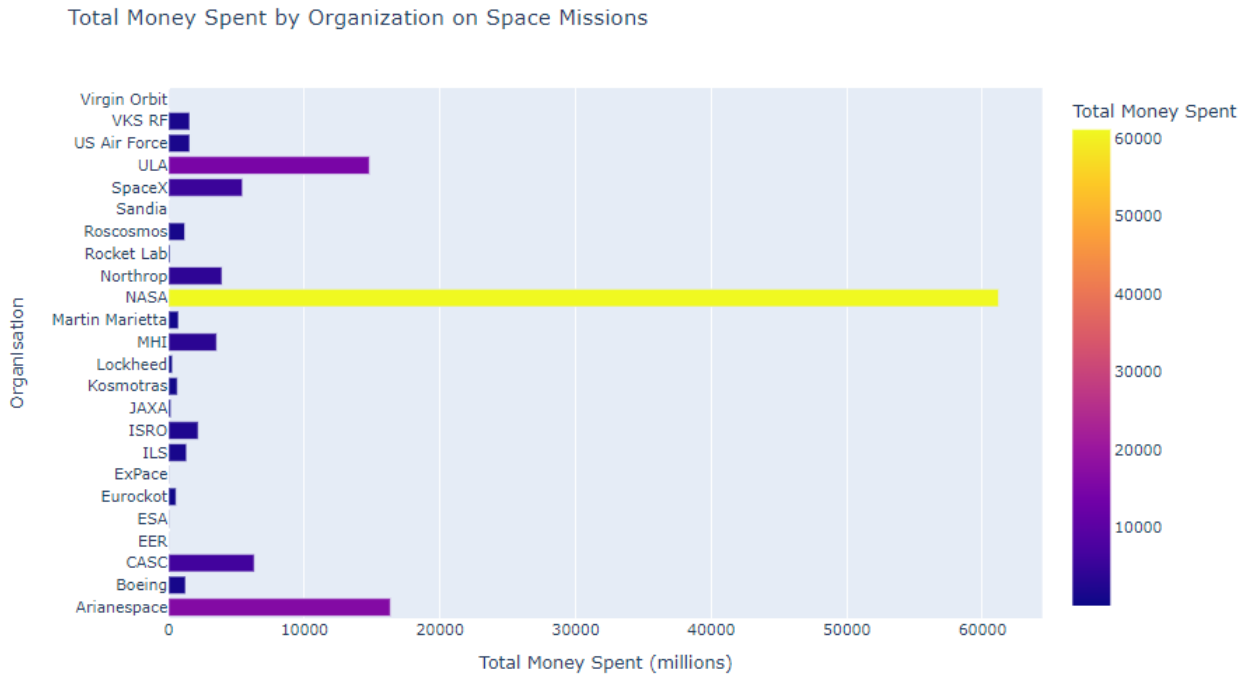
```
fre = df["Mission_Status"].value_counts()
plt.figure(figsize=(10, 2))
fre = fre.sort_values(ascending=False)
fig = px.pie(fre, values=fre.values, names=fre.index, title='Mission Status Distribution')
fig.update_traces(textposition='inside', textinfo='percent+label')
fig.update_layout(width=900)
fig.show()
output_path = os.path.join(output_dir, 'mission_status_df.png')

fig.write_image(output_path, width=1200, height=400, scale=4)
```

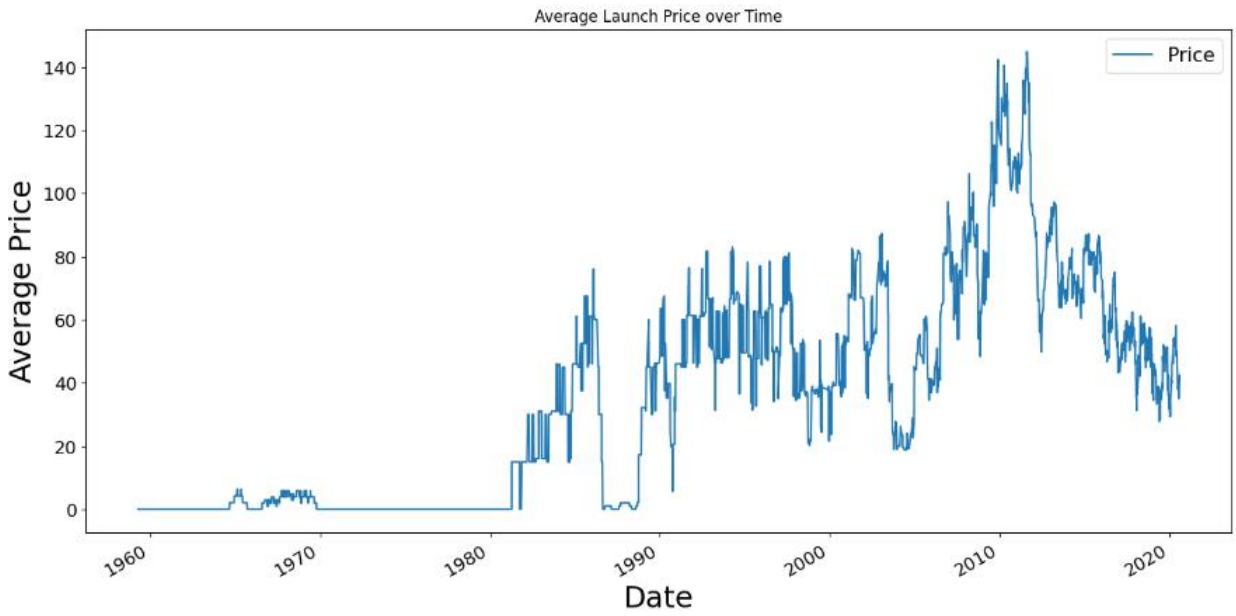


As we can see, 90% of the missions were successful. These figures are impressive and mean that every mission is taken very seriously.

Analyse the Total Amount of Money Spent by Organisation on Space Missions



NASA is at the forefront in terms of the amount of money spent on space missions. NASA spends the most money on space missions because it carries out a wide range of ambitious projects, from exploring other planets to sending astronauts into space.

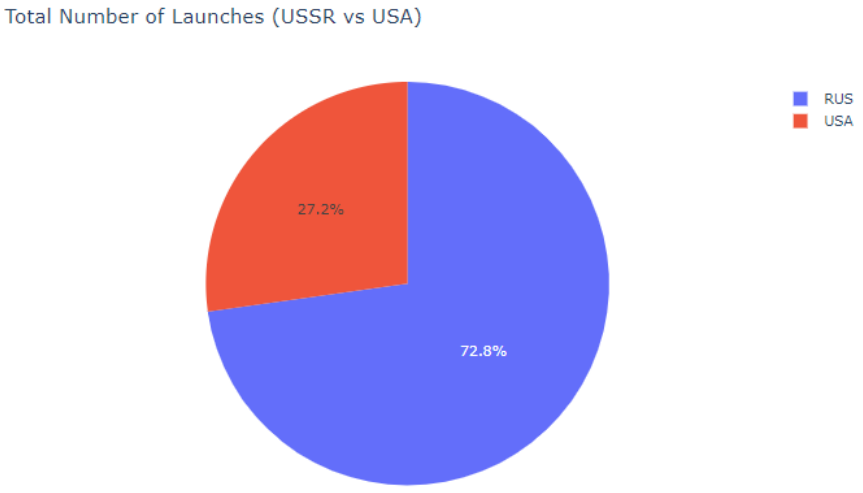


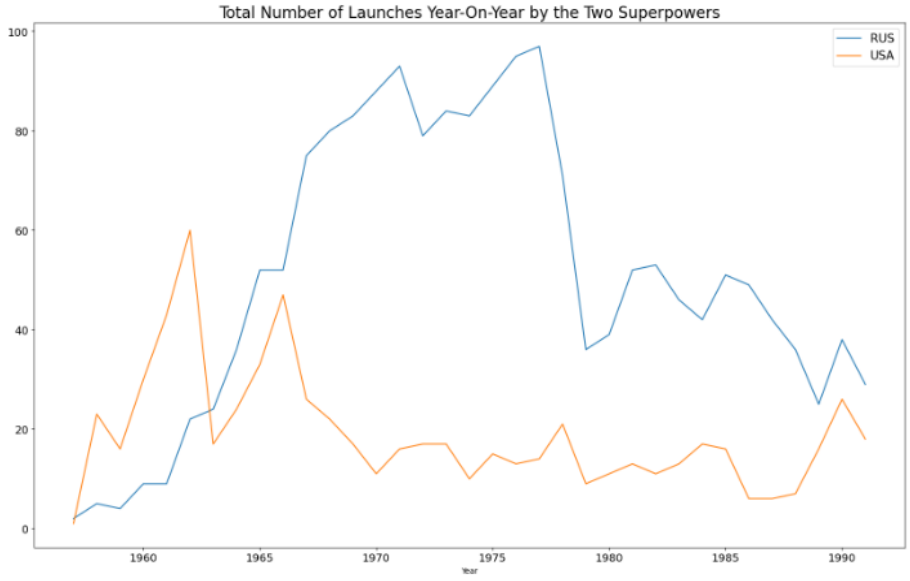
This graph shows how the average prices vary from 1957 to 2020. The beginning of the graph shows exactly 0 because in those years most of the mission's capital was classified, so we replaced those values with 0. The largest peak was during the period of 2010-2014. We also see a huge dip at the end of 1980 and a significant increase at the beginning of 1990.

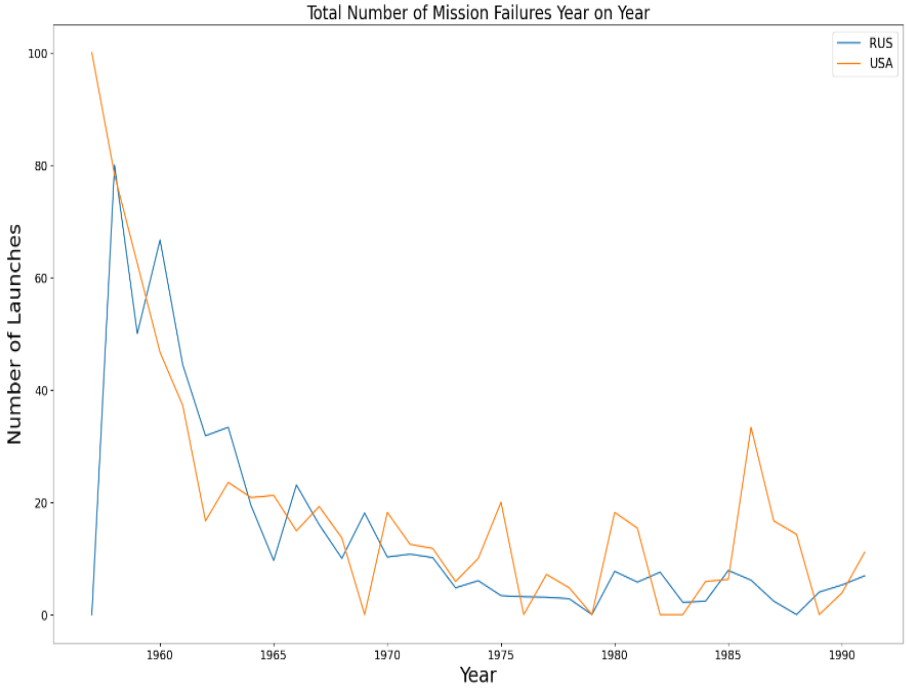
Cold War Space Race: USA vs USSR

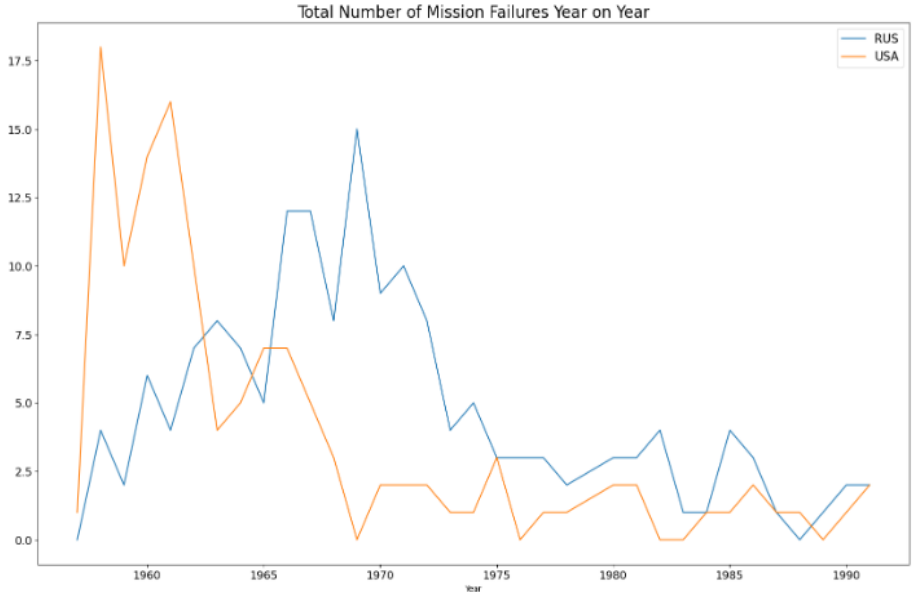
In [48]:
```
fig = px.pie(title = 'Total Number of Launches (USSR vs USA)',
             values=coldwar_df['Country'].value_counts(),
             names=coldwar_df['Country'].value_counts().index)
fig.show()
output_path = os.path.join(output_dir, 'pie.png')

fig.write_image(output_path, width=1200, height=400, scale=4)
```




Total Number of Launches Year-On-Year by the Two Superpowers
RUS
USA
100
80
60
40
20
0
1960
1965
1970
1975
1980
1985
1990
Year


Total Number of Mission Failures Year on Year
RUS
USA
Number of Launches
100
80
60
40
20
0
1960
1965
1970
1975
1980
1985
1990
Year


Total Number of Mission Failures Year on Year
RUS
USA
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
1960
1965
1970
1975
1980
1985
1990
Year