Practical 7

Write a python program to download appropriate dataset and explore random variable, Probability mass function, Probability density function, Cumulative distribution function, Discrete probability distribution and continuous probability distribution using scipy.stats , rv_discrete class and rv_continuous class .

In [2]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import rv_discrete, rv_continuous, norm, poisson
```

Load Dataset

In [9]:
```
df = pd.read_csv("titanic.csv")
df.head()
```

Out[9]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0

Explore Random Variable

In [12]:
```
sibsp_values = df['SibSp'].dropna().astype(int)
sibsp_values.value_counts().sort_index()
```

Out[12]:
```
SibSp
0    608
1    209
2     28
3     16
4     18
5      5
8      7
Name: count, dtype: int64
```

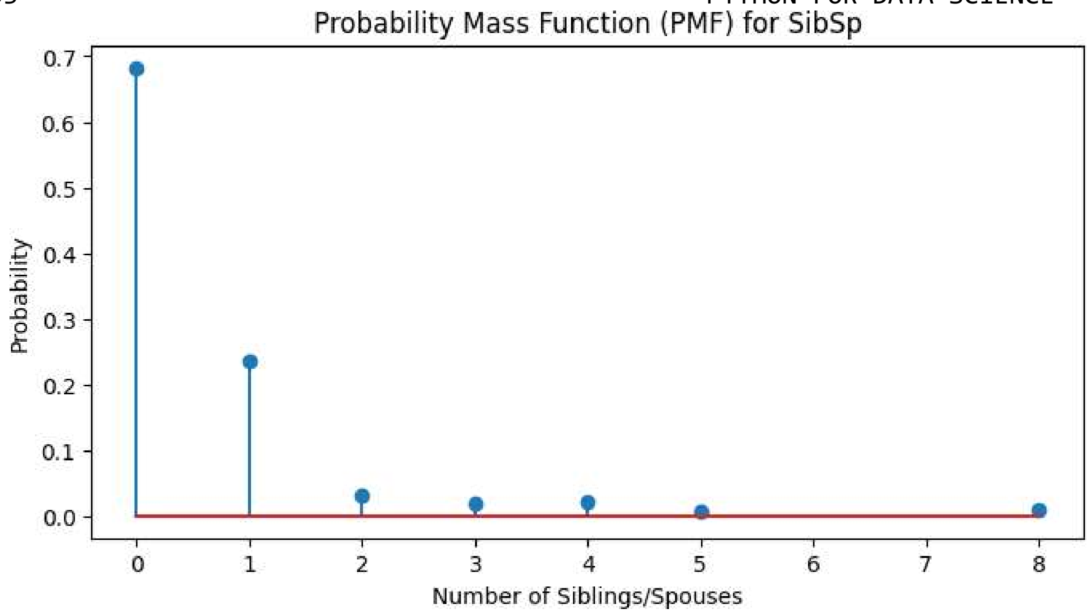
In [15]:
```
ages = df['Age'].dropna()
ages.describe()
```

Out[15]:
```
count    714.000000
mean      29.699118
std       14.526497
min        0.420000
25%       20.125000
50%       28.000000
75%       38.000000
max       80.000000
Name: Age, dtype: float64
```

Probablity Mass function

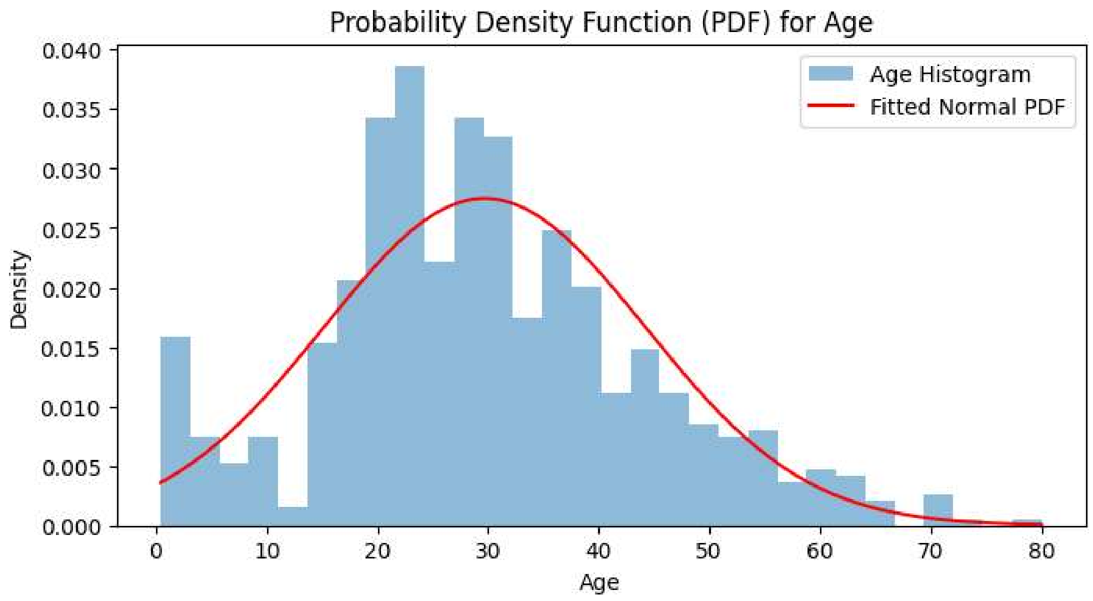
In [22]:
```
values  =  np.array(sibsp_values.value_counts().sort_index().index)
pro = np.array(sibsp_values.value_counts(normalize=True).sort_index())
disrv = rv_discrete(name='custom_sibsp', values=(values, pro))
```

In [23]:
```
plt.figure(figsize=(8,4))
plt.stem(values,  disrv.pmf(values))
plt.title('Probability Mass Function (PMF) for SibSp')
plt.xlabel('Number of Siblings/Spouses')
plt.ylabel('Probability')
plt.show()
```



Probability Density Function

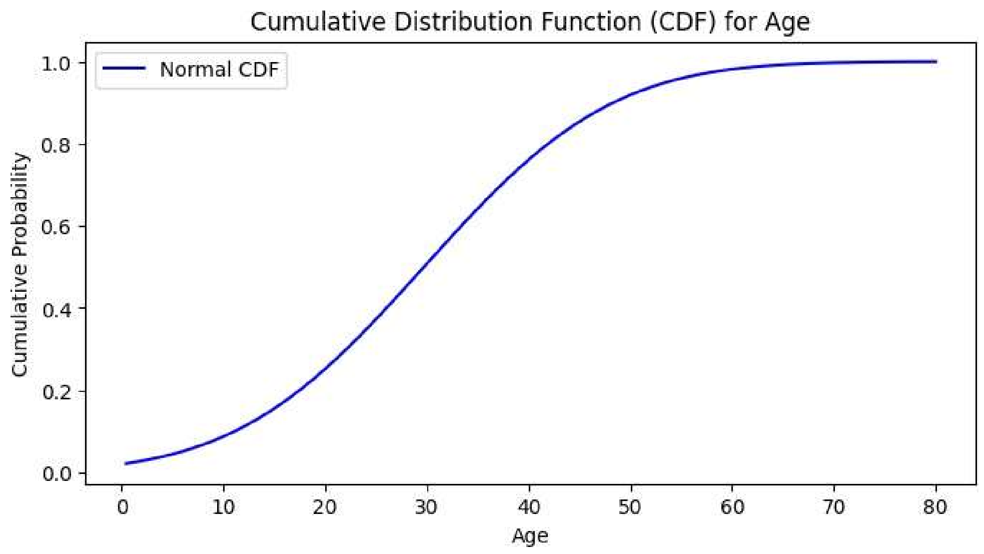
In [27]:
```python
mu, std = norm.fit(ages)
custom_continuous_rv = norm(loc=mu, scale=std)
x = np.linspace(ages.min(), ages.max(), 1000)
plt.figure(figsize=(8,4))
plt.hist(ages, bins=30, density=True, alpha=0.5, label='Age Histogram')
plt.plot(x, custom_continuous_rv.pdf(x), 'r-', label='Fitted Normal PDF')
plt.title('Probability Density Function (PDF) for Age')
plt.xlabel('Age')
plt.ylabel('Density')
plt.legend()
plt.show()
```

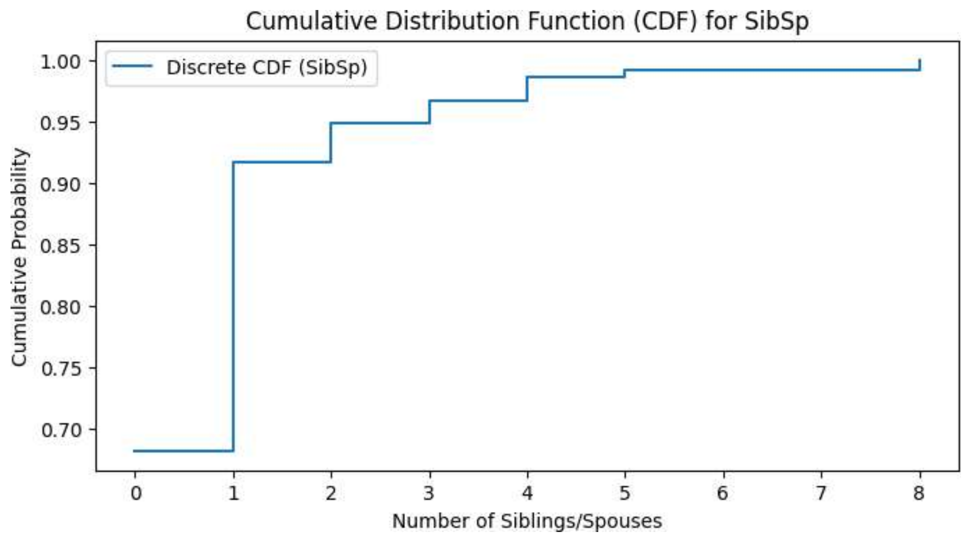


Cumulative Distribution Function (CDF)

In [29]:
```
plt.figure(figsize=(8,4))
plt.plot(x, custom_continuous_rv.cdf(x), 'b-', label='Normal CDF')
plt.title('Cumulative Distribution Function (CDF) for Age')
plt.xlabel('Age')
plt.ylabel('Cumulative Probability')
plt.legend()
plt.show()

plt.figure(figsize=(8,4))
cdf_values = np.cumsum(custom_discrete_rv.pmf(values))
plt.step(values, cdf_values, where='post', label='Discrete CDF (SibSp)')
plt.title('Cumulative Distribution Function (CDF) for SibSp')
plt.xlabel('Number of Siblings/Spouses')
plt.ylabel('Cumulative Probability')
plt.legend()
plt.show()
```



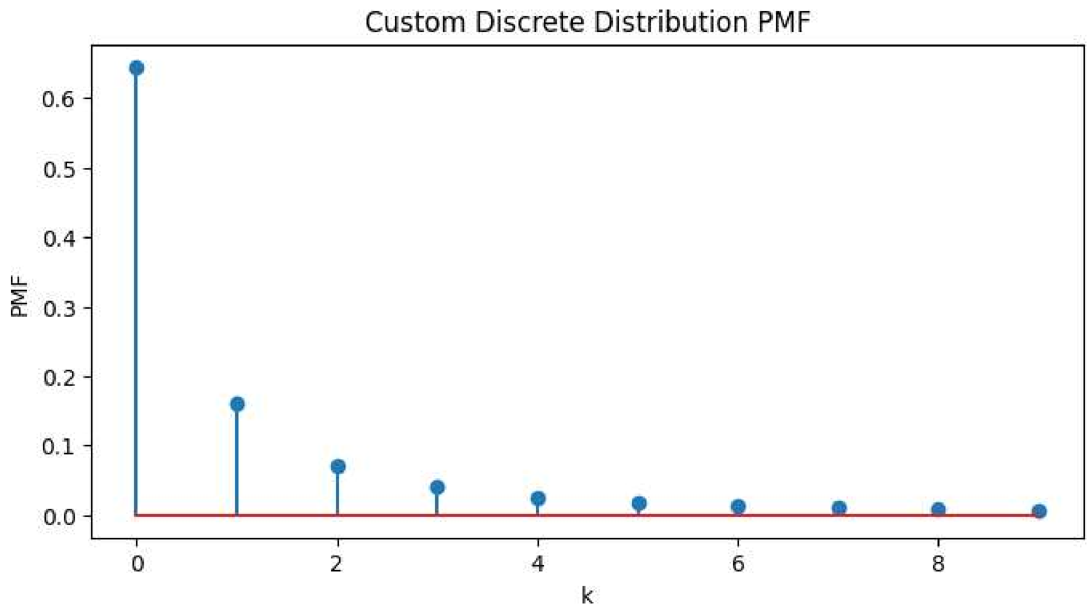


Discrete probability distribution

In [31]:
```
class CustomDiscrete(rv_discrete):
    def _pmf(self, k):
        vals = np.arange(10)
        pmf_vals = 1 / (vals + 1)**2
        pmf_vals = pmf_vals / pmf_vals.sum()
        return np.interp(k, vals, pmf_vals)

custom_discrete = CustomDiscrete(name='custom_discrete')

k = np.arange(10)
plt.figure(figsize=(8,4))
plt.stem(k, custom_discrete.pmf(k))
plt.title('Custom Discrete Distribution PMF')
plt.xlabel('k')
plt.ylabel('PMF')
plt.show()
```

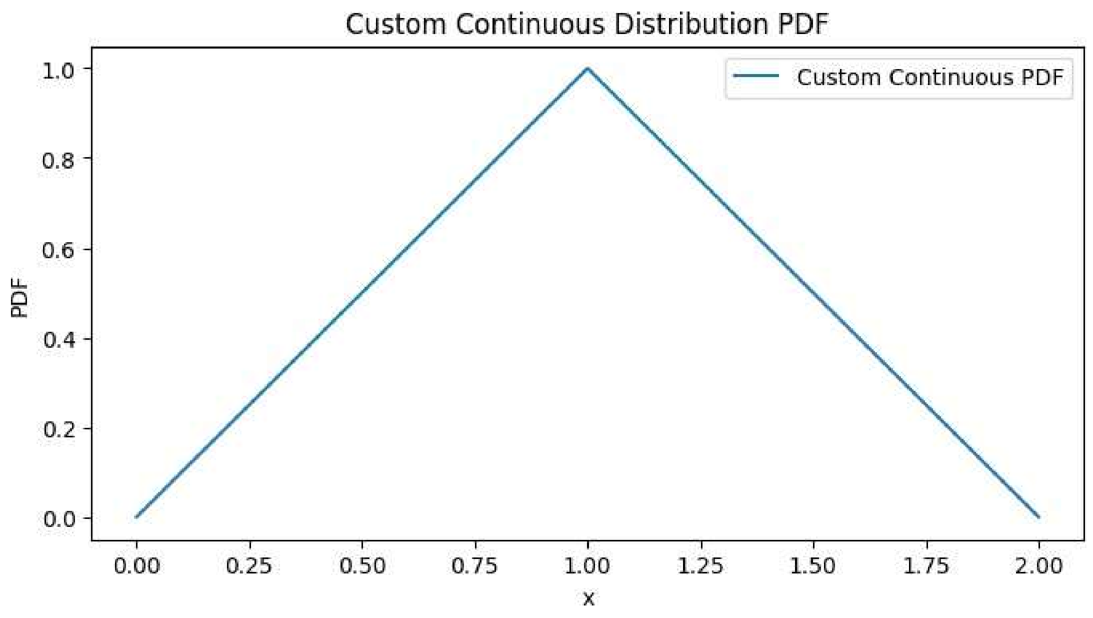


Continuous Probablity Distribution

In [33]:
```
class CustomContinuous(rv_continuous):
    def _pdf(self, x):
        return np.where((x >= 0) & (x <= 1), x, np.where((x > 1) & (x <= 2), 2 -

custom_continuous  = CustomContinuous(name='custom_continuous', a=0, b=2)

x = np.linspace(0, 2, 1000)
plt.figure(figsize=(8,4))
plt.plot(x, custom_continuous.pdf(x), label='Custom Continuous PDF')
plt.title('Custom Continuous Distribution PDF')
plt.xlabel('x')
plt.ylabel('PDF')
plt.legend()
plt.show()
```


Custom Continuous Distribution PDF
Custom Continuous PDF
PDF
x
1.0
0.8
0.6
0.4
0.2
0.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00