In [1]:

```python
import numpy as np
import scipy.stats as stats

# Sample data
data = [10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70]

# Mean (Average)
mean = np.mean(data)
print("Mean:", mean)

print()

# Median (Middle Value)
median = np.median(data)
print("Median:", median)

# Mode (Most Frequent Value)
#mode = stats.mode(data)
#print("Mode:", mode.mode[0][0])
print()

# Quartiles (Dividing Data into Four Parts)
quartiles = np.percentile(data, [25, 50, 75])
print("Quartiles:", quartiles)
print()
```

```
Mean: 40.0

Median: 40.0

Quartiles: [25. 40. 55.]
```

Distributions

Discrete Distribution (Dice Roll)

In [2]:
```
dice_roll = np.random.randint(1, 7, size=1000)
unique_values, counts = np.unique(dice_roll, return_counts=True)
print("Discrete Distribution (Dice Roll):")
for value, count in zip(unique_values, counts):
    print(f"{value}: {count} occurrences")
print()
```

```
Discrete Distribution (Dice Roll):
1: 153 occurrences
2: 169 occurrences
3: 189 occurrences
4: 161 occurrences
5: 160 occurrences
6: 168 occurrences
```

Binomial Distribution

In [3]:
```
n_trials = 10  # Number of trials
p_success = 0.5  # Probability of success
binomial_samples = np.random.binomial(n_trials, p_success, size=1000)
print("Binomial Distribution (n=10, p=0.5):")
```

```
Binomial Distribution (n=10, p=0.5):
```

Normal Distribution

In [4]:
```
mu = 0  # Mean
sigma = 1  # Standard deviation
normal_samples = np.random.normal(mu, sigma, size=1000)
print("Normal Distribution (mu=0, sigma=1):")
#print(normal_samples)
```

```
Normal Distribution (mu=0, sigma=1):
```

Poisson Distribution

In [5]:

```
lambda_parameter = 3  # Mean rate
poisson_samples = np.random.poisson(lambda_parameter, size=1000)
print("Poisson Distribution (lambda=3):")
print(poisson_samples)
```

```
Poisson Distribution (lambda=3):
[0 3 3 1 2 1 8 4 3 5 3 4 5 2 3 8 7 6 3 4 8 0 2 1 8 1 5 4 3 5 2 2 1 4 2 4
1
 5 2 2 2 2 2 3 7 3 1 5 1 3 3 3 2 6 2 1 6 3 7 1 4 1 1 3 2 3 3 5 2 1 3 2 1
3
 1 4 1 5 5 3 0 3 2 2 4 3 3 4 0 2 2 5 4 6 3 3 2 3 3 2 3 8 5 1 4 0 3 1 3 1
3
 3 1 1 3 2 3 2 1 1 5 5 0 3 2 2 1 4 3 5 3 4 4 2 4 2 2 3 3 4 6 2 4 2 3 5 3
1
 7 3 4 3 3 2 1 0 3 5 4 0 4 6 3 7 2 2 0 4 2 1 4 4 5 5 4 1 1 3 3 5 3 4 1 4
6
 2 6 1 6 1 2 1 2 0 4 2 3 3 4 2 2 3 3 2 6 3 3 0 1 3 3 1 4 4 3 0 1 4 2 1 4
2
 1 5 5 5 1 2 4 0 5 4 0 2 6 1 4 2 5 5 5 2 4 4 5 3 1 1 2 3 2 3 2 3 7 2 3 2
6
 5 3 5 6 3 6 3 1 6 4 3 3 0 3 4 3 2 3 4 1 3 2 4 2 4 2 4 4 1 2 5 3 4 2 3 3
7
 4 2 3 3 2 3 3 2 2 1 3 2 1 3 7 2 8 0 3 2 3 8 6 5 4 4 5 5 4 2 2 1 7 4 2 5
2
 2 2 4 3 1 2 5 1 3 1 3 2 5 3 4 5 3 3 3 3 4 4 4 2 2 1 3 7 2 0 4 4 3 2 4 2
3
 3 2 0 2 4 1 1 2 6 3 2 2 2 3 4 3 2 3 1 8 5 2 2 2 1 3 4 3 2 4 3 5 5 4 2 0
0
 3 2 4 5 0 2 4 4 4 3 2 2 4 3 3 3 4 2 1 2 1 6 7 3 3 3 7 2 3 7 0 3 2 2 4 4
5
 1 1 4 9 4 4 4 0 5 1 1 4 1 1 4 3 5 3 3 0 1 7 4 3 2 2 4 2 0 2 2 7 3 4 2 1
3
 7 3 6 5 4 2 6 1 5 1 5 1 2 5 6 2 2 3 4 3 3 3 4 4 3 4 2 5 5 2 6 3 1 3 1 3
7
 4 4 7 2 8 1 3 1 4 5 2 3 3 3 4 1 4 7 1 3 3 2 1 2 6 2 1 8 5 4 3 3 4 3 3 2
0
 3 2 0 3 4 5 3 1 3 0 3 0 1 7 4 3 4 2 2 0 3 1 6 3 4 1 0 4 4 0 3 6 3 3 1 1
5
 3 3 3 1 3 3 1 1 2 4 3 3 2 2 6 2 3 1 2 3 1 1 1 4 3 5 3 6 2 2 6 2 8 6 7 5
5
 1 3 1 2 6 4 3 1 3 2 1 4 5 3 2 4 3 3 4 4 1 2 2 3 3 2 4 3 2 3 1 4 2 3 2 2
2
 4 6 3 5 1 5 3 3 3 1 1 2 1 4 3 0 5 2 4 2 1 3 2 1 5 1 6 1 0 3 2 5 4 5 3 7
2
 4 4 0 2 2 1 3 3 3 3 4 1 7 3 5 0 5 3 3 2 3 4 5 1 3 2 3 3 5 6 2 4 6 3 4 3
3
 4 3 2 2 3 3 0 5 7 6 3 0 1 3 0 1 3 4 6 7 1 2 1 3 4 4 9 2 3 5 5 5 1 5 3 2
0
 2 6 3 3 1 0 2 3 4 3 5 1 3 0 1 2 6 4 2 2 3 4 3 7 3 4 2 4 0 1 4 2 4 2 2 2
6
 2 1 4 3 3 4 3 3 0 6 4 3 1 4 1 4 4 2 2 3 1 3 4 5 3 2 3 3 2 2 4 3 2 5 2 3
2
 4 2 3 2 3 4 4 5 5 4 1 3 1 3 6 4 1 4 5 4 0 3 3 5 5 3 2 3 2 2 3 2 2 4 2 3
1
 5 3 4 6 3 5 3 4 1 3 5 3 5 4 1 2 3 3 1 1 4 3 2 2 1 1 1 3 3 4 2 1 3 1 5 4
2
 4 5 4 2 2 8 3 5 0 2 8 0 4 2 2 1 1 4 2 1 5 4 5 2 1 4 1 1 1 1 3 5 2 2 4 3
2
 3 7 3 4 5 6 4 2 2 5 2 1 8 3 4 0 5 1 1 1 1 4 4 2 1 4 2 2 7 1 2 3 5 2 4 2
5
 6]
```

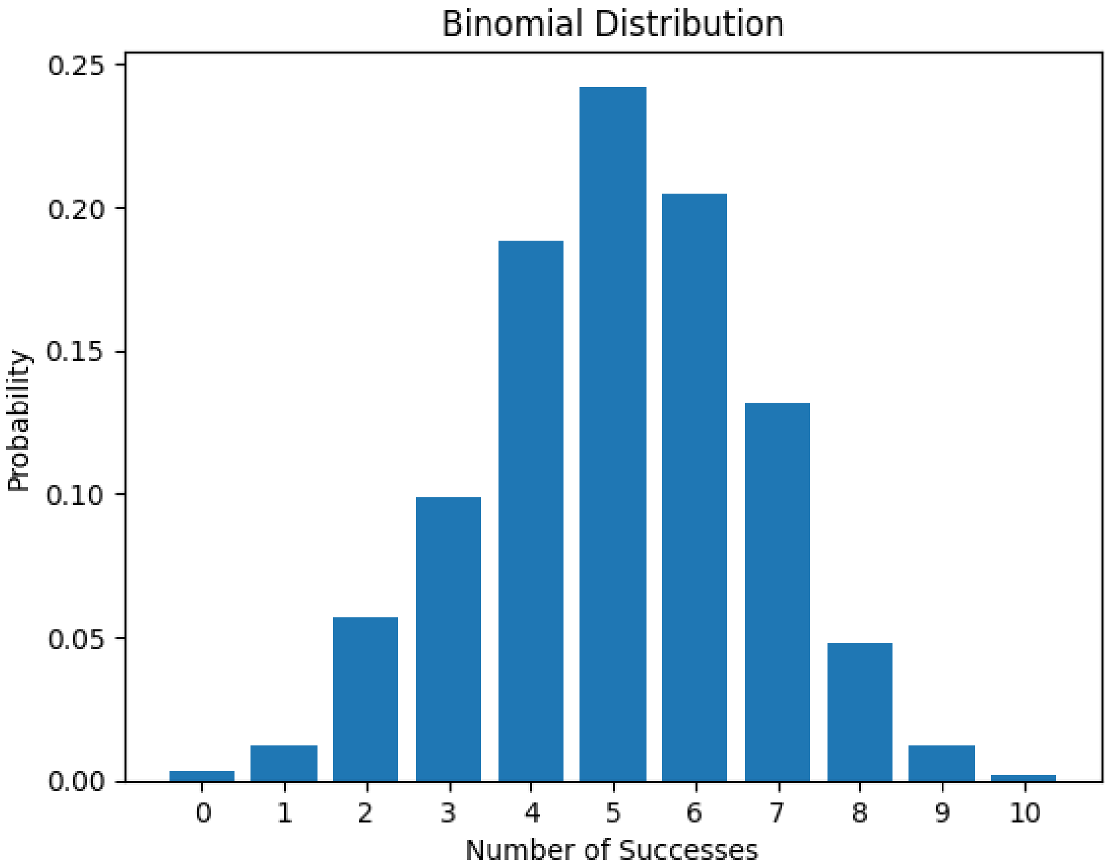
In [18]:

```
print(binomial_samples)
import matplotlib.pyplot as plt

binomial_samples = np.random.binomial(n_trials, p_success, size=1000)

# Create a histogram of the binomial samples
plt.hist(binomial_samples, bins=np.arange(n_trials + 2) - 0.5, rwidth=0.8,
plt.xlabel('Number of Successes')
plt.ylabel('Probability')
plt.title('Binomial Distribution')
plt.xticks(range(n_trials + 1))
plt.show()
```

```
[6 3 5 6 6 5 3 6 3 4 4 6 6 8 6 7 7 4 5 7 2 4 4 7 6 5 5 5 4 3 5 1 6 6 8 4
8
 4 4 6 4 3 4 5 4 3 3 4 6 3 6 4 3 4 2 6 3 4 3 4 6 2 6 5 5 3 6 5 3 5 5 9 3
4
 6 4 2 5 4 5 4 6 5 7 5 4 6 6 4 4 2 8 3 6 5 3 5 7 5 3 5 5 2 7 7 5 4 5 2 4
6
 7 5 3 3 5 6 5 5 3 4 1 2 8 4 7 5 5 5 4 8 8 8 6 4 6 4 8 5 2 6 4 6 5 6 4 4
8
 6 3 6 6 6 6 3 6 7 7 6 3 4 6 3 5 3 9 2 4 3 5 4 6 3 5 6 6 2 5 7 3 5 3 3 4
2
 6 6 5 3 6 7 4 7 5 3 6 5 4 8 6 4 6 5 6 5 4 3 5 7 3 6 4 3 4 3 5 6 8 4 6 6
3
 8 1 7 5 5 3 8 3 7 6 6 4 2 3 2 4 5 5 4 3 6 2 4 5 5 2 6 5 4 5 5 4 5 6 4 9
5
 4 6 6 2 4 5 4 4 5 5 2 5 7 7 5 6 7 2 4 7 6 4 4 5 7 5 3 6 5 8 7 5 6 2 6 4
3
 5 7 8 3 6 6 4 3 5 4 4 5 5 5 5 5 4 4 4 6 4 8 5 4 4 7 6 7 6 6 6 5 5 2 5 4
5
 7 5 6 4 5 5 8 6 8 7 4 4 5 8 4 4 3 7 4 6 7 8 6 5 7 4 6 7 6 6 6 5 5 4 4 7
6
 3 5 4 7 5 6 6 7 6 5 5 4 7 4 6 5 8 3 3 6 7 5 7 6 3 8 4 5 5 6 6 4 3 5 9 6
7
 5 5 4 5 7 1 3 5 4 3 5 7 5 3 8 5 4 4 5 6 5 3 6 7 5 8 2 6 8 3 7 5 4 4 5 7
6
 3 7 2 6 6 6 4 7 8 5 4 5 2 3 4 3 8 4 6 6 7 5 2 6 4 3 5 5 6 7 2 6 5 6 5 3
6
 8 6 6 5 8 4 7 3 5 5 3 6 5 4 5 4 4 7 2 3 8 6 4 6 7 5 5 3 3 4 4 6 3 1 4 4
6
 7 4 3 5 4 1 7 7 5 6 6 3 8 4 5 5 6 5 4 4 7 4 7 6 3 5 7 4 6 2 3 6 7 5 5 2
6
 6 4 6 5 6 4 7 5 3 7 4 4 4 6 5 3 7 5 4 5 6 6 3 3 6 4 8 7 3 3 6 3 7 5 3 4
4
 3 4 5 6 5 6 8 5 5 8 7 5 6 4 5 3 6 5 4 3 6 5 3 5 8 6 7 5 5 5 4 5 4 6 5 6
4
 6 2 7 4 3 6 7 8 7 5 8 4 5 5 5 4 6 6 4 7 7 4 6 5 5 5 7 6 2 5 3 4 4 6 5 4
7
 5 5 0 3 7 4 4 6 6 5 4 3 6 5 4 5 9 3 7 5 5 4 4 7 4 4 5 5 3 5 4 4 5 6 7 5
3
 3 8 6 3 6 7 7 6 7 3 5 6 5 5 5 8 5 5 6 4 2 1 4 3 5 3 2 6 6 7 6 7 7 5 5 3
6
 4 2 9 7 6 1 4 2 5 5 5 3 4 6 4 7 5 8 3 4 4 4 5 7 4 4 5 5 3 4 5 4 3 5 5 4
4
 5 5 8 5 6 7 7 6 4 3 3 6 5 5 6 7 6 5 4 2 6 7 6 4 5 3 5 6 6 8 5 5 5 2 6 6
6
 3 4 7 8 2 4 5 4 3 7 4 5 4 3 6 6 6 5 8 3 3 6 3 6 2 8 4 6 4 3 5 5 6 4 5 4
3
 5 3 2 5 6 6 6 6 6 5 5 5 2 3 8 5 6 5 5 6 6 4 6 5 4 7 4 4 5 5 7 3 4 8 5 3
4
 4 7 5 5 5 6 4 6 5 5 3 6 8 7 5 7 7 6 6 6 6 2 4 6 3 9 5 6 7 7 5 5 4 6 4 5
5
 3 5 4 1 7 4 5 9 3 3 5 2 6 5 5 6 7 5 3 6 6 2 4 6 8 7 8 5 7 5 7 6 9 3 6 6
8
 4 7 8 7 4 3 6 6 4 3 6 6 6 5 2 7 7 4 7 2 5 3 7 7 5 4 5 6 1 7 3 4 7 7 6 8
3
 6]
```



In [20]:

```
print(normal_samples)
```

```
6
  0.88870437  0.13192828 -0.65506421  0.90376434  0.333964    0.5563515
4
 -0.2304618  -0.19241535 -1.40082306  0.14037677 -0.04805252 -1.9475790
1
  0.11840558  0.11577466 -1.71587774 -0.00492178  0.46012878 -0.8715684
4
  1.77940213  1.40244604 -0.8384593   1.90815701 -0.2096806  -0.3522218
4
 -0.88762703  2.90806501 -0.69192822  1.22519015 -0.46146887 -0.4150353
 -1.08289323  2.39316257  1.14695479  0.59979459  0.07806629 -0.0277774
8
 -0.1025012   0.84972768  0.55200667  0.11879818 -0.27914923 -1.0628268
3
 -0.13427666 -0.31213344  0.11075289  0.14059892  1.08264059 -0.3618146
7
 -0.1626485   0.13623918  1.11384608 -1.9336412   1.24243559 -0.3259364
5
  1.32394008 -0.16754965 -0.4537134   0.28270828 -0.26148852  1.8670833
2
```

In [15]:

```python
print(poisson_samples)
```

```
[ 3  5  7  2  5  1  3  3  4  4  1  2  3  2  2  2  1  3  4  2  7  6  4  1
  5  2  1  2  1  5  8  1  3  3  2  3  5  3  6  6  3  3  6  2  0  4  2  1
  5  5  5  3  4  0  2  3  1  3  2  4  2  1  9  2  4  3  3  2  1  4  3  6
  2  6  3  2  2  3  3  3  2  4  3  4  5  2  1  2  1  2  3  3  2  2  1  6
  1  4  2  1  2  4  1  1  2  1  4  5  2  5  2  7  3  0  1  4  4  4  1  3
  3  4  2  3  1  1  4  1  1  2  4  5  2  1  3  1  3  3  3  3  6  2  7  5
  7  3  1  4  3 11  5  2  4  5  7  3  6  6  1  3  3  0  7  1  2  3  4  6
  3  0  2  2  5  4  2  5  0  5  4  3  2  5  1  3  1  4  0  3  4  2  4  0
  3  2  0  8  3  2  2  1  3  2  4  1  3  3  2  4  2  2  4  3  2  4  1  1
  2  4  4  1  1  3  4  1  4  2  5  3  2  3  5  0  1  3  6  2  4  3  3  2
  3  2  3  3  7  5  6  5  3  2  6  4  2  4  1  4  0  1  3  4  3  3  0  2
  5  1  4  2  3  3  3  2  3  3  5  4  3  6  4  2  6  3  2  1  3  1  4  0
  2  2  1  1  2  2  3  1  2  2  5  4  6  4  4  3  2  3  2  1  5  3  2  5
  2  5  5  3  6  5  4  4  4  1  2  4  5  4  2  1  6  5  3  3  6  1  2  2
  3  2  2  3  5  3  2  5  1  3  2  2  3  3  4  3  4  3  7  5  5  2  2  4
  3  4  5  5  2  2  4  2  4  5  1  0  1  3  1  1  3  4  3  5  3  3  2  5
  1  6  2  1  3  5  2  2  3  1  7  4  4  3  0  2  3  6  3  1  5  5  5  6
  3  4  2  3  2  0  4  2  3  2  1  4  3  5  4  1  4  1  4  4  3  2  3  3
  4  1  5  3  3  3  0  5  2  3  5  2  5  5  2  5  2  3  3  3  2  3  2  5
  4  5  2  6  2  4  3  1  3  2  4  1  4  3  6  8  6  3  1  4  3  7  3  6
  5  3  5  0  3  7  6  3  3  3  2  5  4  1  2  2  4  5  3  5  3  1  2  4
  5  5  4  2  2  2  0  4  3  3  4  3  4  2  1  3  7  4  5  6  4  1  0  1
  0  2  7  6  5  2  0  4  2  7  2  4  3  4  3  5  7  5  2  3  6  1  2  6
  2  4  6  4  6  2  5  4  3  4  1  4  3  1  0  3  2  3  4  4  4  1  1  5
  3  4  4  2  2  1  3  4  4  6  3  2  3  2  3  4  3  4  6  5  4  4  1  4
  4  3  2  2  1  3  3  2  4  4  4  3  2  3  1  3  4  3  4  4  6  1  3  3
  2  3  7  1  1  0  4  2  2  4  5  7  4  5  1  2  2  2  1  1  5  6  2  3
  2  4  1  1  3  1  3  0  2  6  0  1  0  3  5  3  2  8  4  4  2  0  1  7
  2  3  4  2  5  1  3  2  3  3  4  3  4  4  5  3  4  6  5  3  3  5  5  6
  4  3  4  4  4  1  5  0  3  5  6  2  3  4  4  6  2  3  3  4  2  3  3  4
  5  1  2  3  3  6  4  3  3  2  1  3  3  1  2  3  4  1  5  1  2  3  2  3
  3  2  4  2  5  1  3  0  3  0  1  2  8  3  4  3  1  4  3  4  3  0  5  2
  1  5  3  1  6  2  4  5  3  6  2  5  1  0  5  3  2  0  2  3  3  3  1  4
  2  2  3  4  3  7  2  2  5  2  1  4  4  4  0  4  4  1  3  1  4  3  0  3
  0  7  0  3  5  4  1  1  3  1  2  3  4  3  3  0  7  3  0  1  2  5  2  3
  1  4  2  1  2  1  5  1  3  2  3  2  3  2  2  0  3  3  2  1  3  4  1  2
  5  3  2  1  1  1  3  5  2  0  1  5  2  1  2  1  3  5  2  2  3  3  2  2
  5  4  3  3  1  7  2  4  2  5  3  3  4  3  0  6  5  7  2  4  3  4  0  1
  5  1  3  9  3  2  3  5  4  4  3  1  2  2  4  3  0  1  2  6  5  4  3  4
  1  2  5  3  2  1  4  1  5  2  3  3  6  5  4  4  6  3  0  1  4  2  2  0
  3  3  6  3  6  3  3  1  2  3  3  0  5  2  5  4  5  0  4  3  3  4  2  4
  0  5  4  0  2  2  1  4  4  2  6  4  2  0  1  4]
```