DISCRETE DISTRIBUTIONS

In [5]:

```
# for inline plots in jupyter
%matplotlib inline
# import matplotlib
import matplotlib.pyplot as plt
# for latex equations
from IPython.display import Math, Latex
# for displaying images
from IPython.core.display import Image
import numpy as np
```

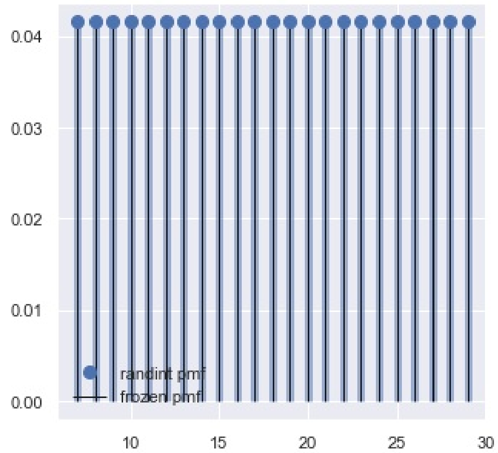
In [6]:

```
# import seaborn
import seaborn as sns
# settings for seaborn plotting style
sns.set(color_codes=True)
# settings for seaborn plot sizes
sns.set(rc={'figure.figsize':(5,5)})
```

UNIFORM DISTRIBUTION

In [9]:

```
from scipy.stats import randint
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1, 1)

# Calculate a few first moments:

low, high = 7, 31
mean, var, skew, kurt = randint.stats(low, high, moments='mvsk')

# Display the probability mass function (``pmf``):

x = np.arange(randint.ppf(0.01, low, high),
              randint.ppf(0.99, low, high))
ax.plot(x, randint.pmf(x, low, high), 'bo', ms=8, label='randint pmf')
ax.vlines(x, 0, randint.pmf(x, low, high), colors='b', lw=5, alpha=0.5)

# Alternatively, the distribution object can be called (as a function)
# to fix the shape and location. This returns a "frozen" RV object holding
# the given parameters fixed.

# Freeze the distribution and display the frozen ``pmf``:

rv = randint(low, high)
ax.vlines(x, 0, rv.pmf(x), colors='k', linestyles='-', lw=1,  label='frozen pmf')
ax.legend(loc='best', frameon=False)
plt.show()

# Check accuracy of ``cdf`` and ``ppf``:

prob = randint.cdf(x, low, high)
np.allclose(x, randint.ppf(prob, low, high))
# True

# Generate random numbers:

r = randint.rvs(low, high, size=1000)
```



You can use Seaborn’s distplot to plot the histogram of the distribution you just created. Seaborn’s distplot takes in multiple arguments to customize the plot. You first create a plot object ax. Here, you can specify the number of bins in the histogram, specify the color of the histogram and specify density plot option with kde and linewidth option with hist_kws. You can also set labels for x and y axis using the xlabel and ylabel arguments.

Bernoulli Distribution

$$P(x) = \begin{cases} 1 - p, & x = 0 \\ p, & x = 1 \end{cases}$$

In [ ]:

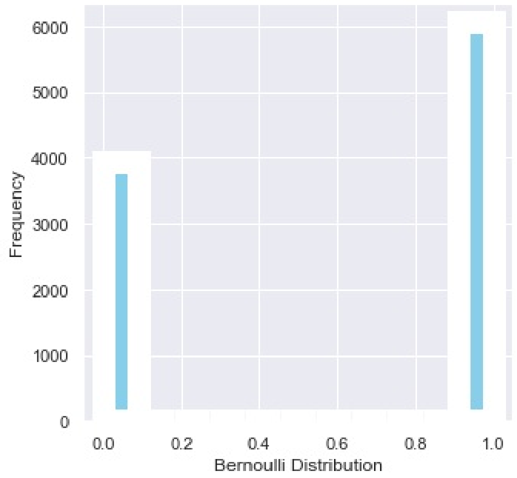
```
from scipy.stats import bernoulli
data_bern = bernoulli.rvs(size=10000,p=0.6)
```

In [17]:

```
ax= sns.distplot(data_bern,
                 kde=False,
                 color="skyblue",
                 hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Bernoulli Distribution', ylabel='Frequency')
```

Out[17]:

```
[Text(0, 0.5, 'Frequency'), Text(0.5, 0, 'Bernoulli Distribution')]
```



BINOMINAL DISTRIBUTION

$$P(x) = \frac{n!}{(n-x)!\,x!} p^x q^{n-x}$$

In [14]:

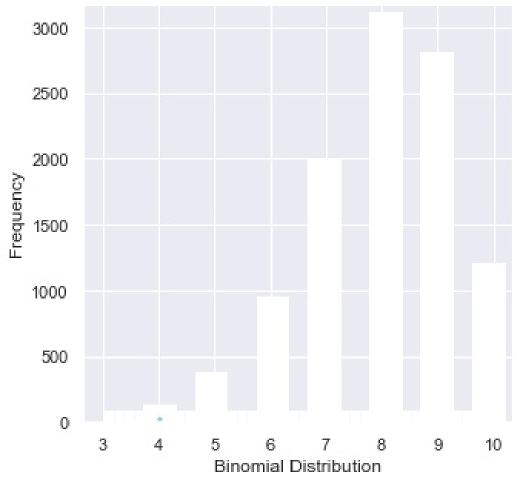
```
from scipy.stats import binom
data_binom = binom.rvs(n=10,p=0.8,size=10000)
```

In [15]:

```
ax = sns.distplot(data_binom,
                  kde=False,
                  color='skyblue',
                  hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Binomial Distribution', ylabel='Frequency')
```

Out[15]:

```
[Text(0, 0.5, 'Frequency'), Text(0.5, 0, 'Binomial Distribution')]
```



Poisson Distribution

Poisson random variable is typically used to model the number of times an event happened in a time interval

$$P(X = x) = e^{-\mu}\frac{\mu^x}{x!} \qquad\qquad for\ x\ = 0, 1, 2, \ldots\ldots$$

In [12]:

```
from scipy.stats import poisson
data_poisson = poisson.rvs(mu=3, size=10000)
```

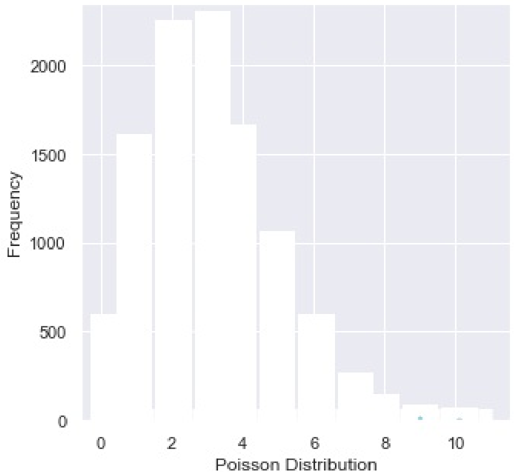
You can generate a poisson distributed discrete random variable using scipy.stats module's poisson.rvs() method which takes μ as a shape parameter and is nothing but the λ in the equation. To shift distribution use the loc parameter. size decides the number of random variates in the distribution. If you want to maintain reproducibility, include a random_state argument assigned to a number.

In [13]:

```
ax = sns.distplot(data_poisson,
                  bins=30,
                  kde=False,
                  color='skyblue',
                  hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Poisson Distribution', ylabel='Frequency')
```

Out[13]:

```
[Text(0, 0.5, 'Frequency'), Text(0.5, 0, 'Poisson Distribution')]
```



In [ ]: