```
#PMF and CDF of Binomial r.v
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats



showsup_probability = 0.9


# create 1000 points from a X ~ Binomial(n=110,p=0.9)
showsup_distribution =  stats.binom(n = 110, p= showsup_probability)

showsup_data = showsup_distribution.rvs(1000)



type(showsup_distribution)
```

```
    scipy.stats._distn_infrastructure.rv_frozen
```

```
showsup_data[:100]
```

```
    array([100, 101, 101, 101, 103, 103,  91, 104,  95,  98, 101,  96,  95,
            99,  98, 101, 102,  95,  95, 102,  97,  96,  98, 101, 100, 101,
            98, 103, 102,  98,  94, 102, 102, 100,  94, 101, 102, 103, 100,
            97,  98, 105,  99,  96, 103,  93,  96,  98,  99, 101,  95, 103,
            98,  96,  94,  98, 101,  96, 102, 101,  98,  96,  93, 102,  98,
           101, 100, 102, 101, 106,  97, 103, 102, 103,  99, 101, 102, 100,
           101,  93,  98, 102,  96, 100, 100, 102,  90, 100, 103,  98,  98,
           101, 100, 102,  98, 102,  99,  93,  98, 103])
```

```
#PMF of data
sns.histplot(showsup_data, bins=18, kde=True)
plt.xlabel("Number of passangers showed up")
plt.grid()
plt.show()
```

```
#CDF:
x = np.linspace(80, 115)

cdf = showsup_distribution.cdf(x)

plt.plot(x, cdf)
plt.xlabel("X-values")
plt.ylabel("Probabilities")
plt.title("CDF of binomial distribution")
plt.grid()
plt.show()
```

```
## T-Shirt sizes problem


id = "13pAHLZl95sprFndmbklG6YZsSHRbA6YA"
print("https://drive.google.com/uc?export=download&id=" + id)
```

```
https://drive.google.com/uc?export=download&id=13pAHLZl95sprFndmbklG6YZsSHRbA6
```

```
!wget "https://drive.google.com/uc?export=download&id=13pAHLZl95sprFndmbklG6YZsSHRb
```

```
--2022-03-31 16:03:28--  https://drive.google.com/uc?export=download&id=13pAHI
Resolving drive.google.com (drive.google.com)... 74.125.195.100, 74.125.195.13
Connecting to drive.google.com (drive.google.com)|74.125.195.100|:443... conne
HTTP request sent, awaiting response... 303 See Other
Location: https://doc-0g-14-docs.googleusercontent.com/docs/securesc/ha0ro937c
Warning: wildcards not supported in HTTP.
--2022-03-31 16:03:29--  https://doc-0g-14-docs.googleusercontent.com/docs/sec
Resolving doc-0g-14-docs.googleusercontent.com (doc-0g-14-docs.googleuserconte
Connecting to doc-0g-14-docs.googleusercontent.com (doc-0g-14-docs.googleuserc
HTTP request sent, awaiting response... 200 OK
Length: 76911 (75K) [text/csv]
Saving to: ‘employees.csv’

employees.csv       100%[===================>]  75.11K  --.-KB/s    in 0.001s
```

```
2022-03-31 16:03:29 (83.3 MB/s) - ‘employees.csv’ saved [76911/76911]
```

```
employees = pd.read_csv("employees.csv")
```

```
employees.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   Age               1470 non-null   int64 
 1   Department        1470 non-null   object
 2   DistanceFromHome  1470 non-null   int64 
 3   Education         1470 non-null   int64 
 4   EmployeeNumber    1470 non-null   int64 
 5   Gender            1470 non-null   object
 6   Height            1470 non-null   int64 
 7   MaritalStatus     1470 non-null   object
 8   MonthlyIncome     1470 non-null   int64 
dtypes: int64(6), object(3)
memory usage: 103.5+ KB
```

```
employees.head()
```

```
employees['Height'].plot(kind='hist', bins=50) # change with the bins value to show
plt.grid()
plt.show()
```

```
# lets get mean and std-dev from the data since we dont know population mean and st

# ASSUMPTION: sample mean and std-dev are good approximations of popoulation means

employees['Height'].mean()
```

```
164.6734693877551
```

```
employees['Height'].std()
```

```
6.887961959078209
```

```
height_dist = stats.norm(loc=employees["Height"].mean(), scale = employees["Height"

sns.histplot(height_dist.rvs(10000), kde=True , stat='density')
```

```
x = np.arange(120,200)

# Plotting pdf using SCIPY
pdf = height_dist.pdf(x) #scipy plotting
plt.plot(x, pdf)
plt.xlabel("Heights")
plt.ylabel("Probabilities")
plt.title("PDF of Normal distribution")
plt.grid()
plt.show()
```

```
height_dist.cdf(x = 150)   # P(X<=150)
```

```
0.016573163101179463
```

```
height_dist.cdf(x = 200)    # P(X<=200)
```

```
0.9999998541520864
```

```
below_150 = height_dist.cdf(x= 150)
below_160 = height_dist.cdf(x= 160)
```

```
import math
math.ceil((below_160 - below_150)*10000)
```

```
2322
```