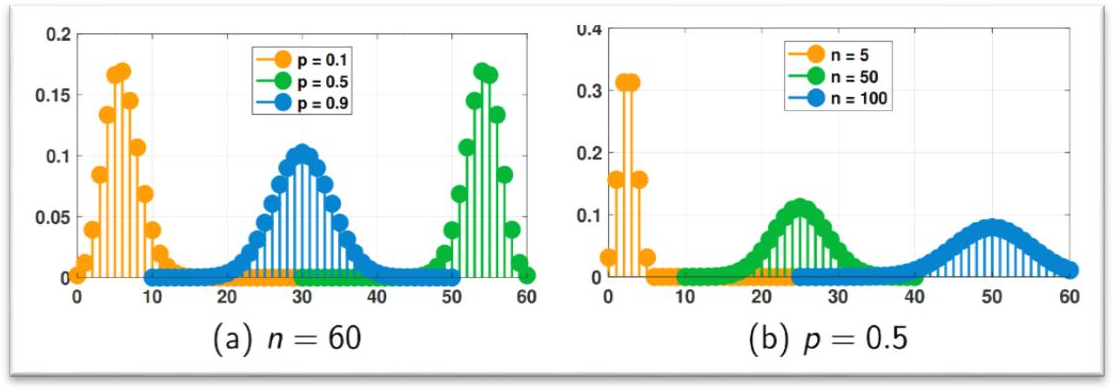
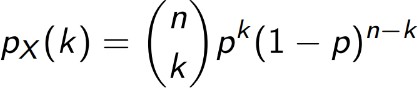
# Binomial Discrete Distribution

### Definition:

A binomial distribution is a discrete probability distribution that models the number of successes in a fixed number of independent Bernoulli trials, where each trial has the same probability of success, denoted by "p." The distribution is characterized by two parameters: the number of trials, denoted by "n," and the probability of success, denoted by "p."

### PMF Formula:



Where:

n : is number of trials

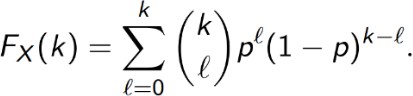
p : probability of success in one trial k: number of successes

### Properties of Uniform distribution:

Mean(E(x))= np E(x²)= np(np+(1-p))

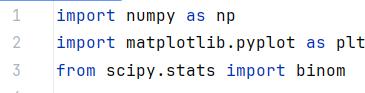
Var(x)= np(1-p)

*CDF:*



### The Code:

We start by importing the necessary libraries:

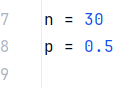


Line 1: **numpy** library helps in mathematical and logical operations on arrays

Line 2 : we use **matplotlib** library for used for plotting the pmf and cdf

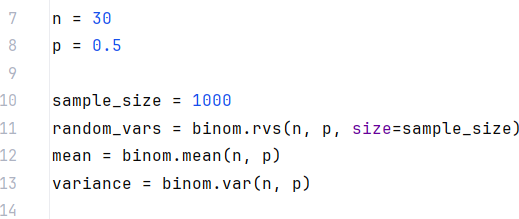
Line 3 : the **scipy.stats** library used for statistical and probabilistic operations

After that we set the parameters of the Binomial Distribution

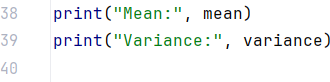


N is number of trials while p is probability of success note that p should be a value between 0 and 1

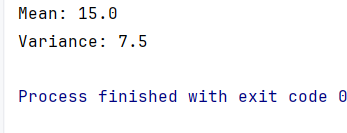
After that we generate the Mean, Variance and random variables



The binom.rvs() , binom.mean() and binom.var() functions are associated with the **scipy.stats** library and used for calcu;ating mean and variance and random variable using n and p ‘as in the context of the code’

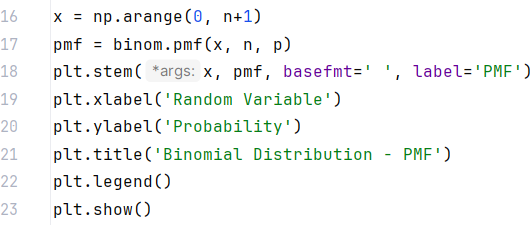
We use the following code to print mean and variance:

The output:



After all that we will plot the PMF and CDF:

**PDF:**



For the previous code:

-np.arange() This function creates an array x using **numpy's** `arange` function. The array contains values from 0 to n

-binom.pmf() this function calculates the PMF values using X array and n and p which are previously initiated

-plt.stem function used to create the stem plot pf PMF x array is the x axis While pmf array is the y axis , we used **basefmt** to clear the base line

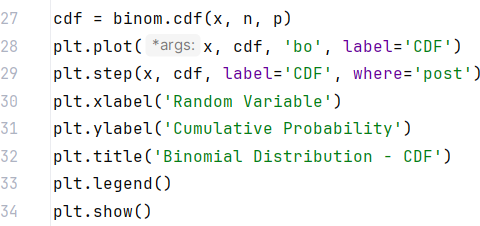
- plt.xlabel() and plt.ylabel() sets the labels of x-axis and y-axis

-plt.title() displays the title of the plot

-plt.legend() displays the legend of the plot

-plt.show() displays the plot on the screen

**CDF:**



for plotting CDF we will do the same as in PMFand just create a new array ‘cdf’

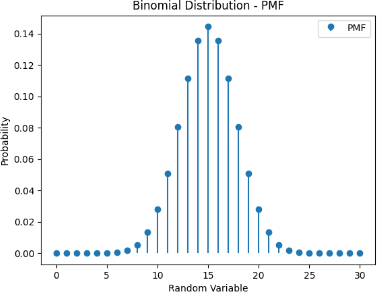
-binom.cdf() to create array ‘cdf’ and calculating CDF values by using x and n and p

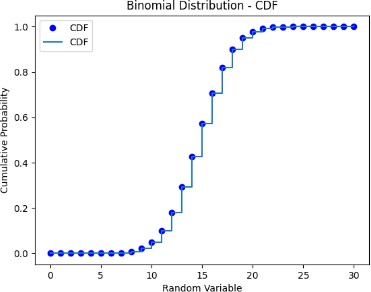
-plt.plot() to plot CDF as dot by using (‘bo’) we may not use it and just use plt.step()

-plt.step() to plot CDF as a step

The last 5 lines are the same as the ones used in plotting PMF

**Plotting Output:**





Definition:

Bernoulli random variable

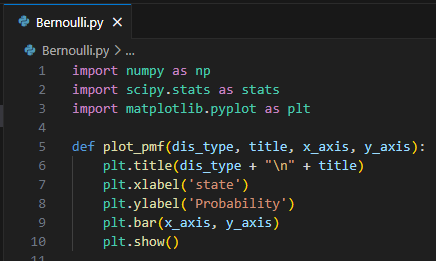
A Bernoulli random variable is a discrete random variable that takes on only two possible values, typically represented as "success" or "failure," or 1 or 0. It is a simple type of random variable that is used to model situations where there are only two possible outcomes.

Px(0) = 1 – P Px(1) = P

Properties of Bernoulli random variable

* Expectaion: E [x] = P
* Second moment: E [x2] = P
* Variance: Var [x] = P(1-P)

**First, we need to import needed libraries and define the plotting function:**

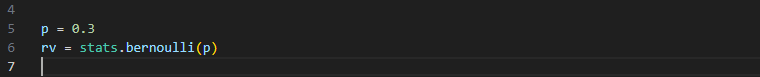


**"numpy"** as **np** for numerical operations, **"scipy,stats"** as **stats** for statistical functions, **"matplotlib.pyplot"** as **plt** for graphing the functions.

For defining plotting function named **"plot\_pmf"** and determining it's parameters from "dis\_type", "title", "x\_axis", "y\_axis"

And creating bar plot using **"plt.bar"** and showing it using **"plt.show"**

#### Second, setting up the Bernoulli distribution:



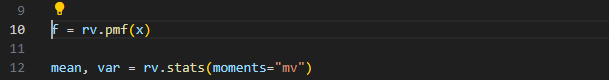
Where **'P'** is the probability of success of the Bernoulli distrbution

In addition, **'stats.bernoulli (p)'** creates a Bernoulli random variable distribution with specified probability **'P'**.

#### Third, Defining values for X (Possible outcomes):

**'np.linespace (0, 1, 2)'** generates an array of two values between 0 and 1 In this case, **X** will be an array of [0 , 1].

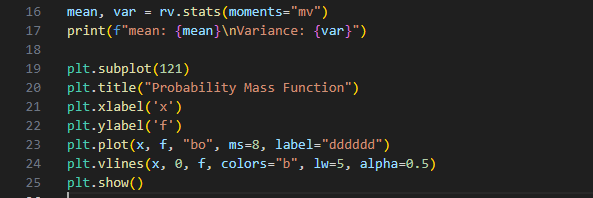
#### Fourth, calculating probability mass function (PFM), Mean and Variance:



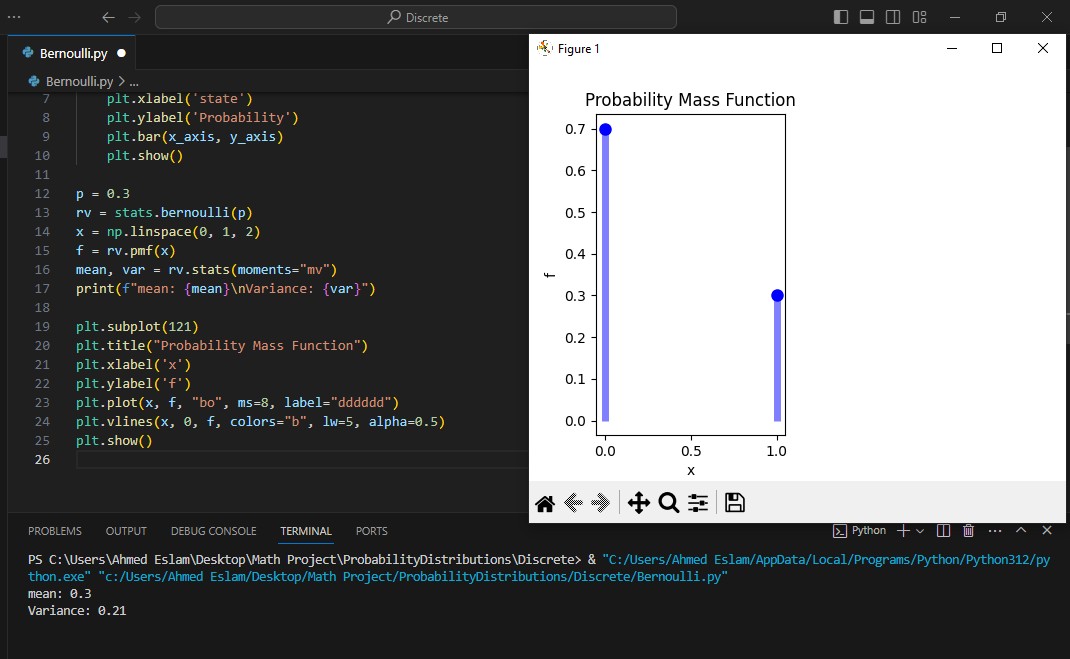
Here **'rv.pmf(x)'** calculates the probability mass function for the specified outcomes in **'X'**

Function **'rv.stats (moments="mv")'**calculates mean and var of Bernoulli

#### Finally, printing mean and variance also, plotting the PMF:



**Output:**



Definition:

## Geometric random variable

The Geometric random variable is a discrete random variable function that is used when one is modelling a series of experiments that have one of two possible outcomes – success or failure – 1 or 0.

Px (k) = (1 – P )k-1 P

Properties of Geometric random variable:

* E[x] = 1

𝑃

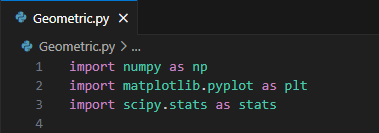
* E[x2] = 2 − 1

𝑃^2 𝑃

* Var[x] = 1−𝑃

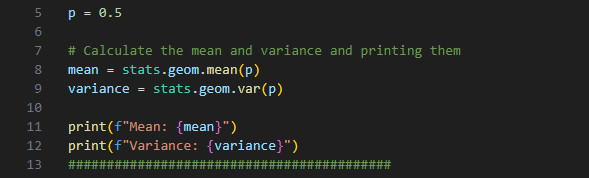
𝑃^2

**First: Importing needed libiraries:**



**"numpy"** as **np** for numerical operations, **"scipy,stats"** as **stats** for statistical functions, **"matplotlib.pyplot"** as **plt** for graphing the functions.

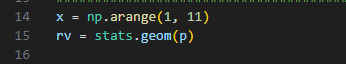
#### Second: Defining the probability and calculating Mean and Variance



Here "p" has 0.5 value that represent probability of success in geometric distribution

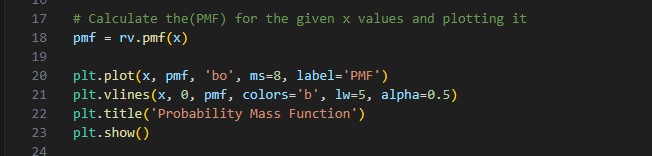
Moreover, using "stats.geom.mean" and "stats.geom.var" from "scipy" libirary to calculate Mean and Variance for "p"

#### Third: Creating values for x and Geometric Distribution

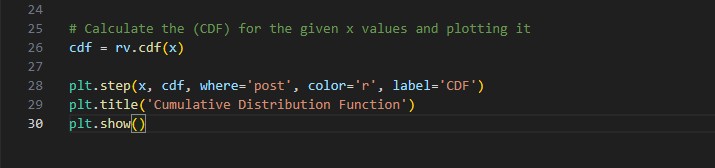


Here "x" is an array from 1 to 10 and "rv" is a Geometric Distribution for probability 'p'.

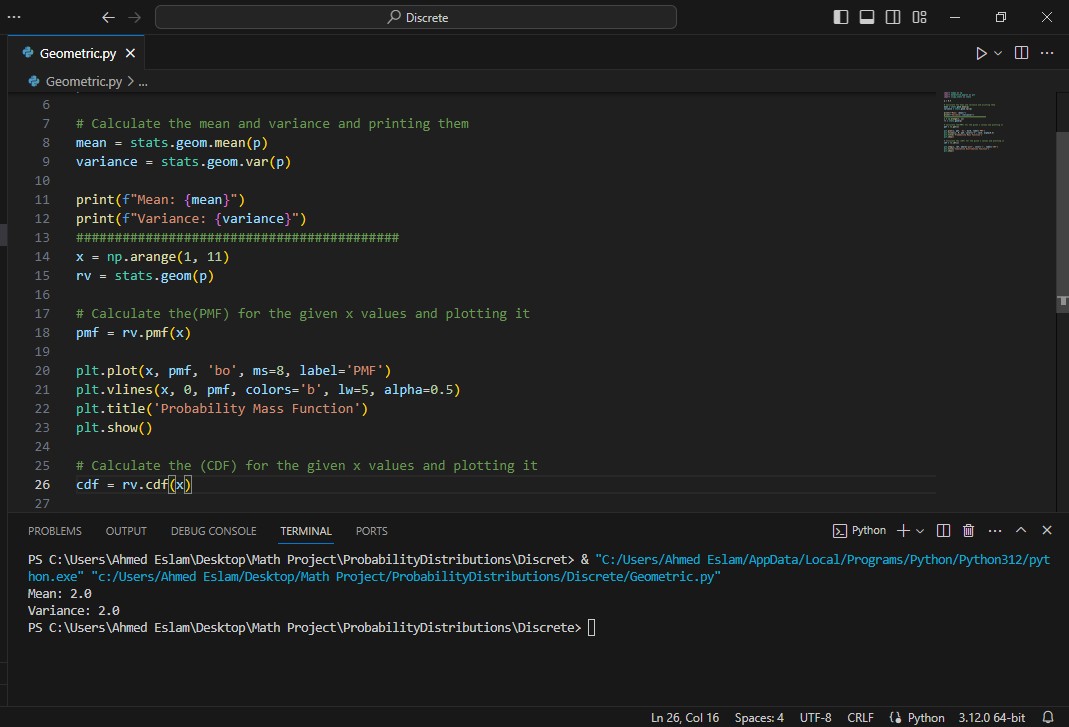
**Fourth: Calculating Probability mass function for "x" and plotting it**



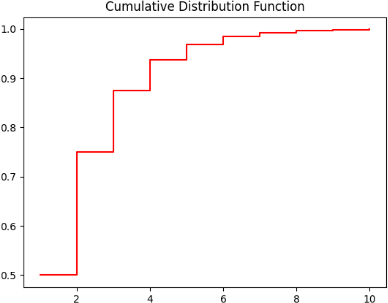
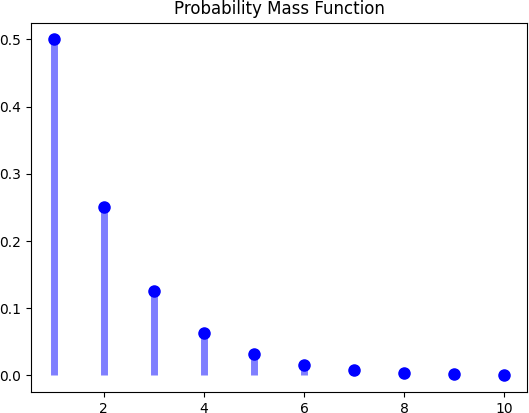
**Fifth: Calculating cummulative density function for "x" and plotting it**

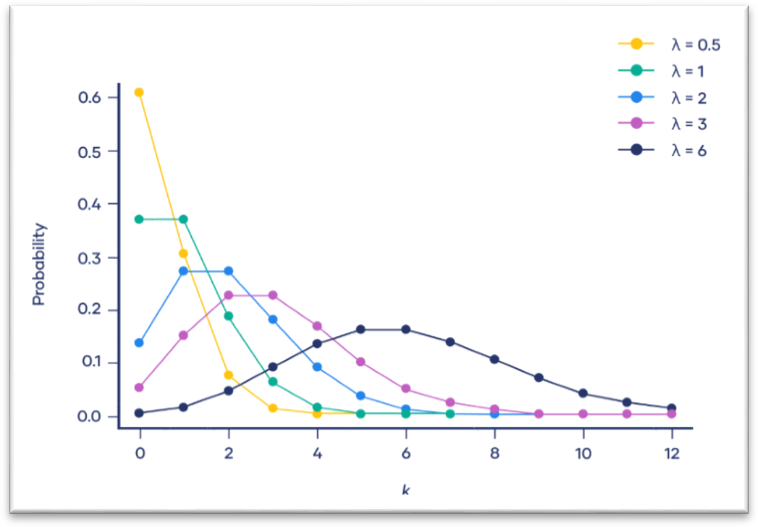


**Output:**



Mean and Variance



**Poisson Distribution:**

1. **Definition**: Is a discrete probability distribution used to model the number of occurrences of a random event in a fixed interval of time or space.
2. **Formula**

P(X = k) = \dfrac{e^{-\lambda} \lambda^k}{k!}Poisson distribution formula is:

Where:

* + Xis a random variable following a Poisson distribution
  + kis the number of times an event occurs
  + P (X = k) Is the probability that an event will occur k times
  + eis Euler’s constant (approximately 2.718)
  + \lambdais the average number of times an event occurs
  + ! is the factorial function

1. **Properties of Poisson Distribution**

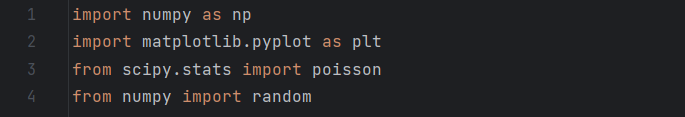
**X ~ Poisson ( 𝜆 )**

1. **Mean:** 𝐸(𝑥) = 𝜆

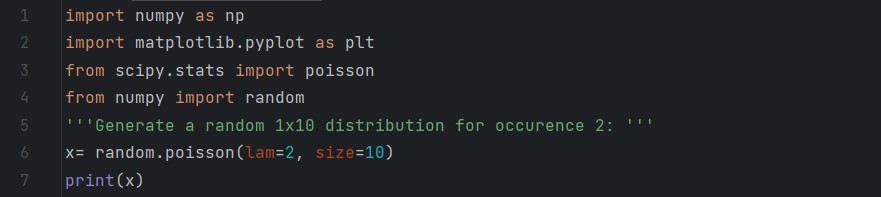
2. **𝐸 (**𝒙 𝟐**)** = 𝜆 + 𝜆2

3. **variance**: 𝑉𝑎𝑟 (𝑥) = 𝜆

1. **Code**

* First, we start by importing the required libraries.
  + Now, to calculate Poisson distribution we need 2 parameters which are 𝜆 & k. In our code, 𝜆 is 𝜇 or our expectation. We need to generate random variable for occurrence 2, we use Poisson’s function which is **“random.Poisson( )”** takes 𝜆 = 2 & 𝑘 = 10.

Then, the code produces a different set of random numbers each time it is executed.

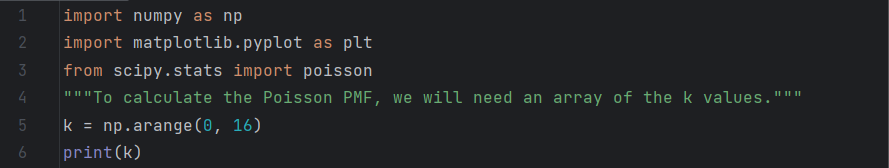


* Output:



* + Here, we need an array of the **k** values. We use **“np.arange ( )”** function and gave it numbers from 0 to 16 and print it.

create an array with these values:



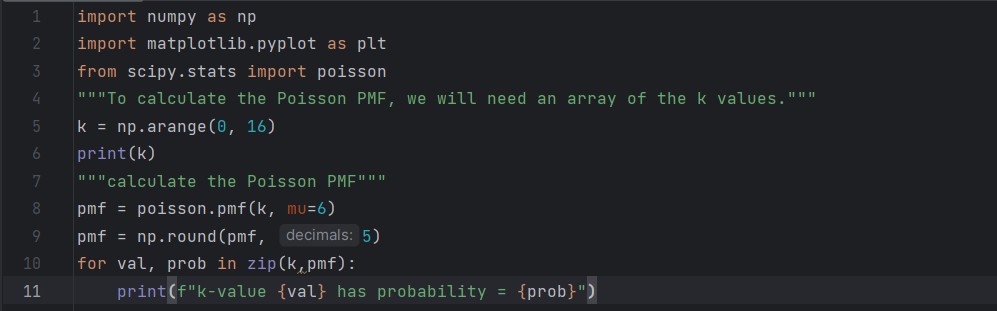
* Output:



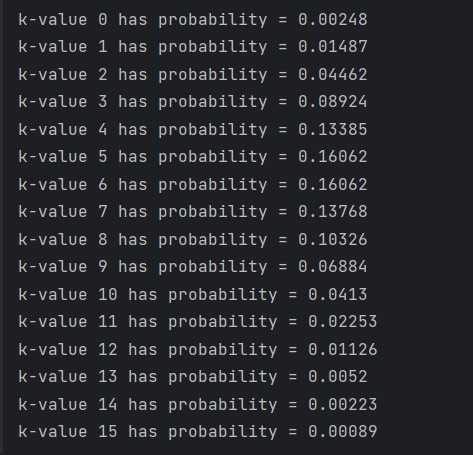
* + to calculate the Poisson PMF, we will use the **“poisson.pmf ( )”** method of the “[**Scipy.poisson**](https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.poisson.html)**”** generator. It will need two parameters:
* k value (the **k** array that we created).
* μ value (which we will set to μ=6 as in our example).

**“np.round ( )” function :** to print 5 digits following the PMF's decimal.

We make **for loop** takes 2 parameters which are the value and the probability, then after substitute in Poisson’s equation, it will give me the probability of each value.



* Output:



* + We will need the **k** values array that we created earlier as well as

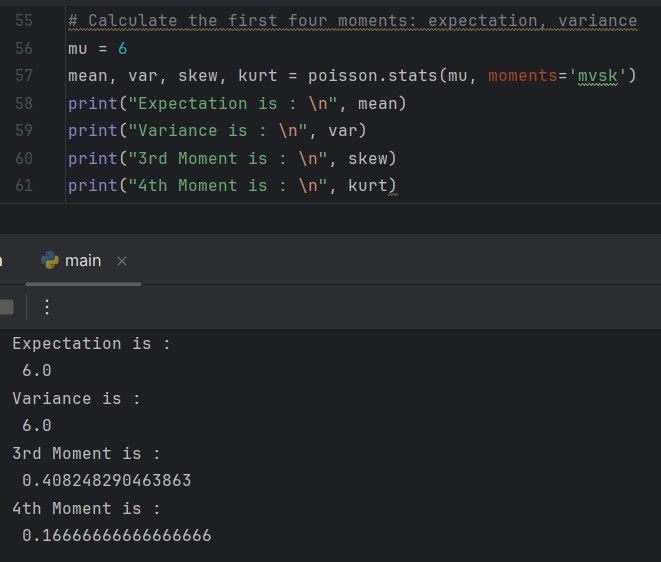
the **PMF** values array in this step. Using **matplotlib library**. We need to plot PMF graph so, we use **“plt.bar( )”** function takes PMF value and k .Then, we add label to X-axis and Y-axis by using **“plt.xlabel ( )”** and **“plt.ylabel ( )”.**

* The same steps as PMF exactly except the function which is

**“poisson.cdf ()”.**

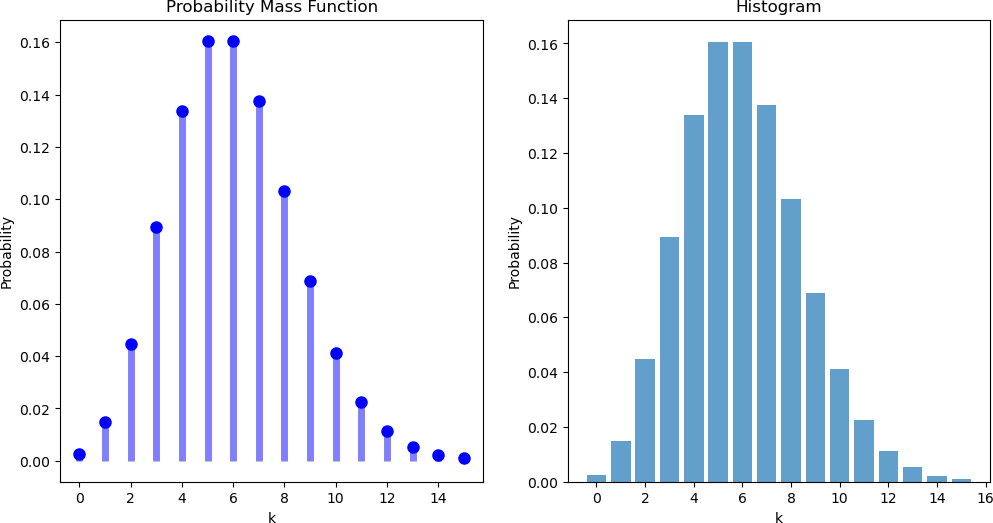


* + In this code, we need to calculate the first four moment.
    - Our first moment is the expectation, we put it constant (μ=6) from the first code.
    - Second moment is the variance which is equal to the expectation. Then 3𝑟𝑑and 4𝑡ℎ moment.

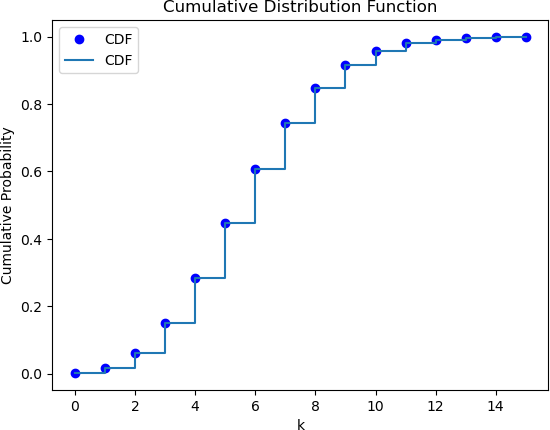


**Results**

By the above codes, finally we can be able to calculate the expectation and variance of PMF and CDF and plot the histogram of both.



(Poisson PMF)

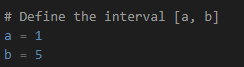


(Poisson CDF)

1. **Import necessary libraries:**



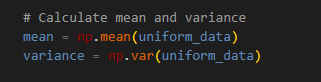
1. **Define the interval [a, b]:**



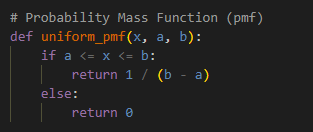
1. **Generate 1000 uniform random variables within the interval [a, b]:**



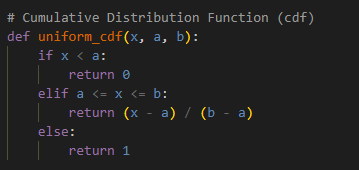
1. **Calculate the mean and variance of the generated data:**



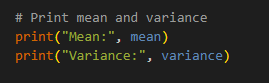
1. **Define the probability mass function (pmf) for a uniform distribution:**



1. **Define the cumulative distribution function (cdf) for a uniform distribution:**



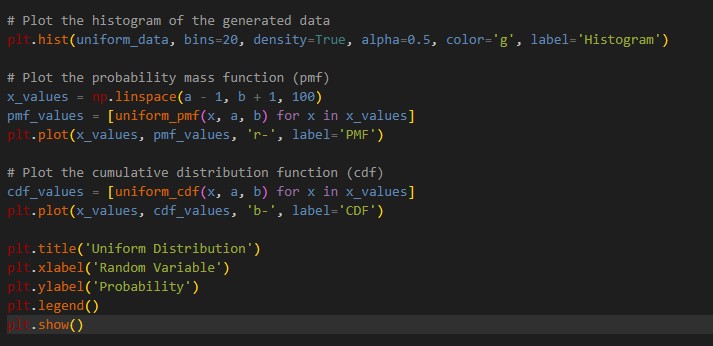
1. **Print the mean and variance:**



**<<sample output :**



1. **Plot the histogram of the generated data along with the PMF and CDF:**

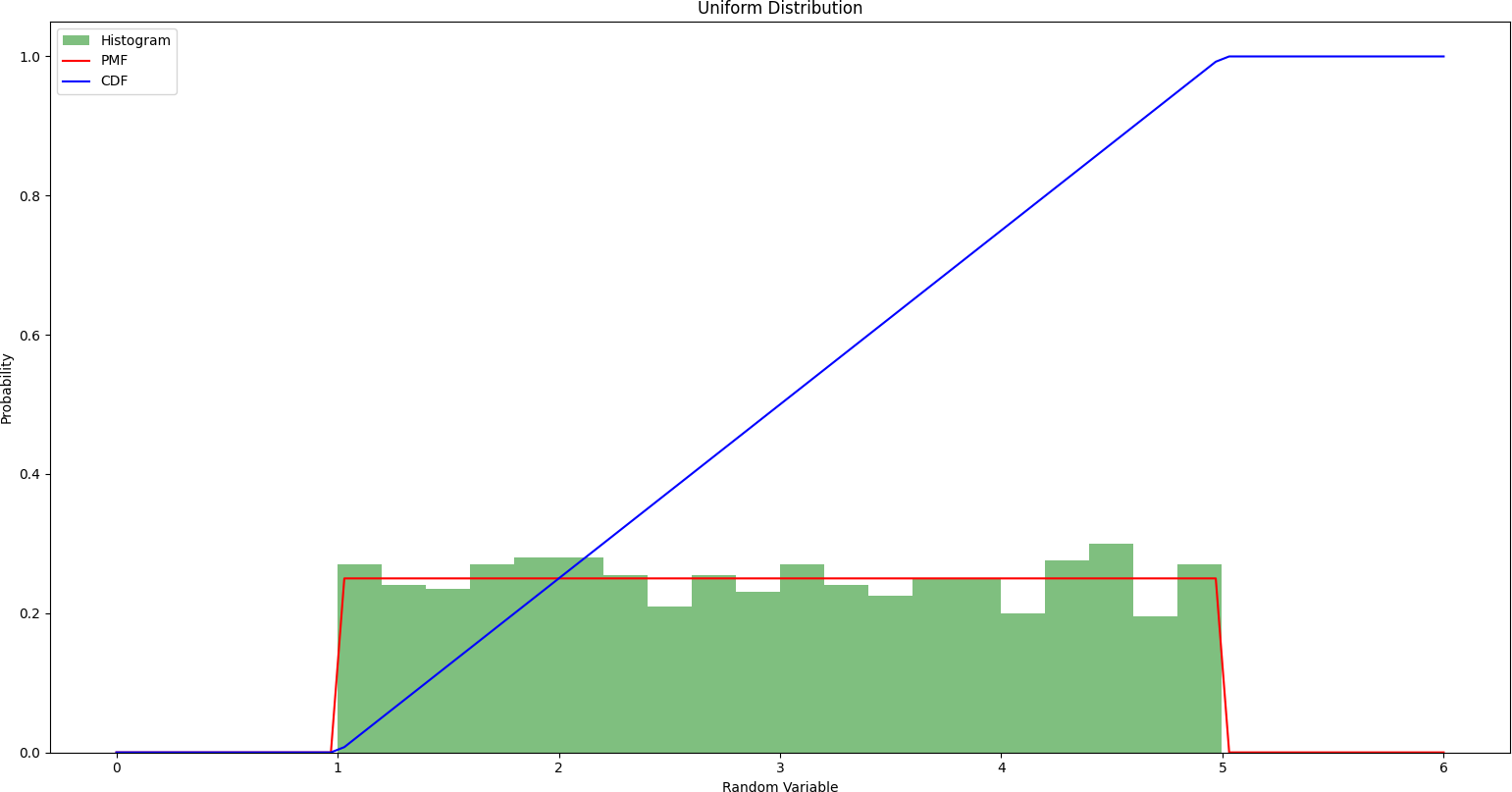


* + The histogram is created using plt.hist with 20 bins, normalized to form a probability density, and plotted in green.
  + The PMF and CDF are calculated and plotted using the

uniform\_pmf and uniform\_cdf functions, respectively. The PMF is shown in red, and the CDF is shown in blue.

* + Finally, the title, labels, and legend are added to the plot, and the plot is displayed using plt.show().

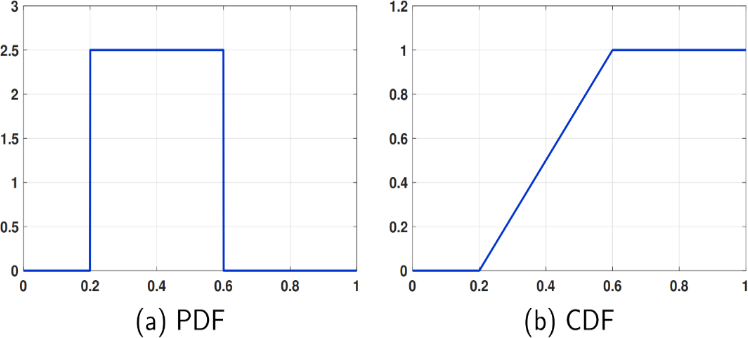
**<<sample output:**



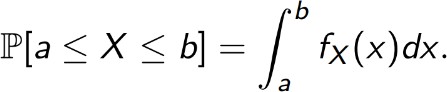
# Uniform Continous Distribution

### Definition:

The uniform distribution is a symmetric probability distribution where all outcomes have an equal probability of occurring. All values in the distribution have a constant probability, making them uniformly distributed.



### Probability Formula:



Where:

a : the begining of interval b : the end of the interval

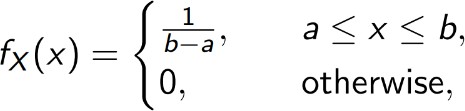
fx(x): is the probability density function”PDF”

### Properties of Uniform distribution:

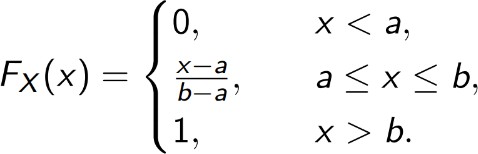




**PDF:**

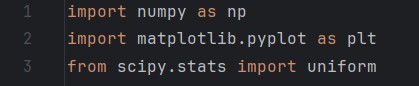


**CDF:**



### The code:

First we import the libraries we need for executing the code :



Line 1: **numpy** library helps in mathematical and logical operations on arrays

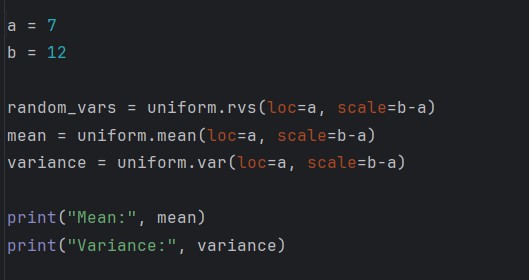
Line 2 : we use **matplotlib** library for used for plotting the pdf and cdf

Line 3 : the **scipy.stats** library used for statistical and probabilistic operations

To calculate and plot the uniform distribution we should set two parameters :

a is the lower bound b is the upper bound

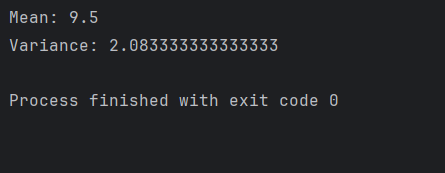
and then generate the random variable , the mean and the variance



The uniform.rvs() , uniform.mean() and uniform.var() functions are associated with the **scipy.stats** library

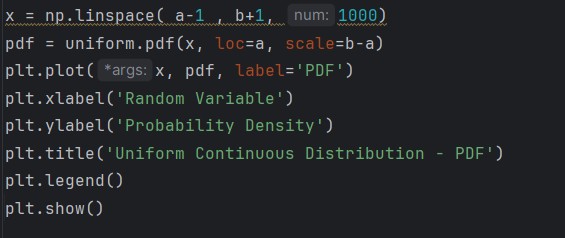
The context of these functions contains (loc=a) which determines that the starting point is a and (scale=b-a) determines the width of the distribution

The output of printing Mean and Variance:



After calculating mean and variance and generating the random variables we will plot PDF and CDF using the following codes:

For PDF:



-The np.linespace( ) function generates an array name x contains 1000 evenly distributed values spaced between a-1 and b+1

-the uniform.pdf() function used to calculate the PDF of the distribution of array x

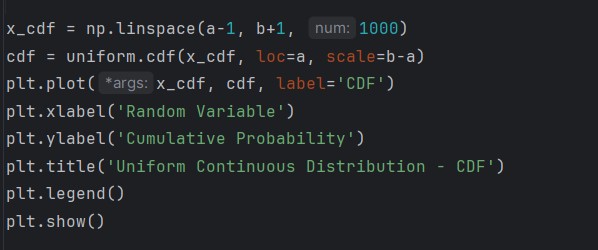
* plt.plot() plots the PDF using x array values as x-axis and pdf values as y-axis
* plt.xlabel() and plt.ylabel() sets the labels of x-axis and y-axis

-plt.title() displays the title of the plot

-plt.legend() displays the legend of the plot

-plt.show() displays the plot on the screen

For CDF:



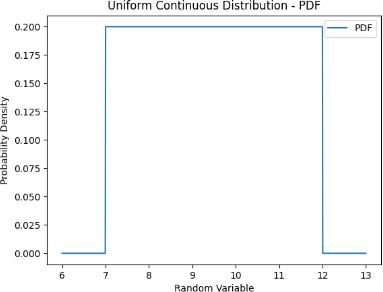
-The np.linespace( ) function generates an array name x\_cdf contains 1000 evenly distributed values spaced between a-1 and b+1 and this array will be used as x-axis in plotting

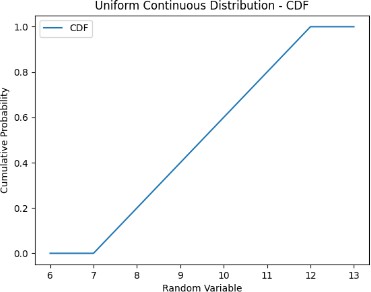
-the uniform.pdf() function used to calculate the PDF of the distribution of array x\_cdf

* plt.plot() plots the CDF using x\_cdf array values as x-axis and cdf array values as y-axis

The next 5 lines function is the same as in PDF plotting

The output of plotting :

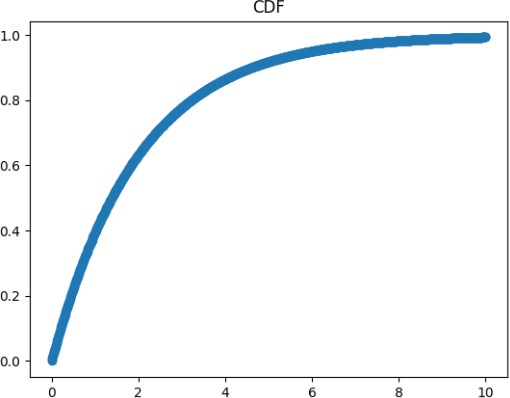
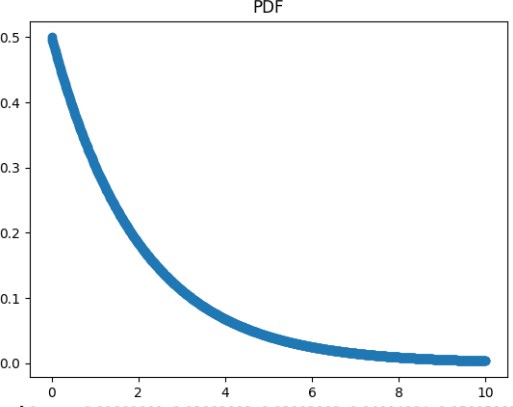




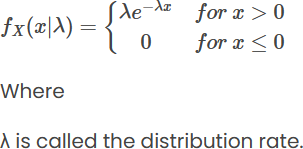
# Exponential Continuous Distribution

### Definition:

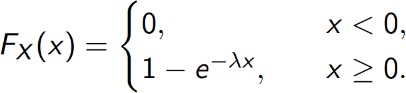
The Exponential distribution is a continuous probability distribution that often concerns the amount of time until some specific event happens. It is a process in which events happen continuously and independently at a constant average rate.



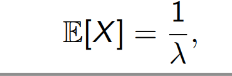
PDF:



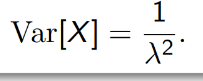
CDF:



Mean:

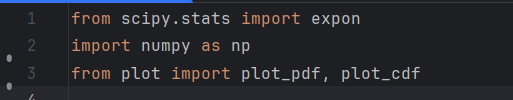


Variance:



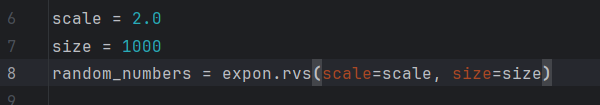
*The code:*

* + The code imports the necessary libraries and modules for performing statistical analysis on data following an exponential distribution.

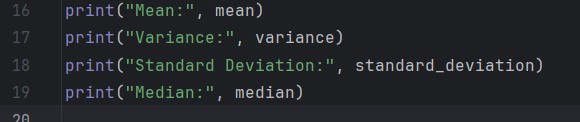


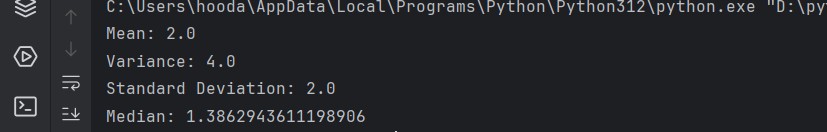
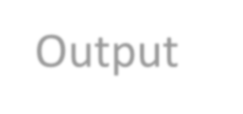
* + Now, we set Random numbers it’s size 1000

Exponential distribution using expon .rvs() function which scale is set to 2.0 ,which is 𝜆 value is 1/scale=0.5



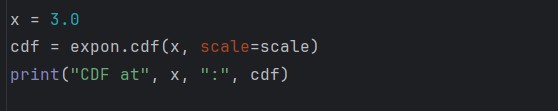
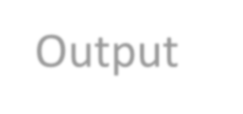
* + Calculating statistics such as mean , variance, standard deviation and median using function from expon class.





Output

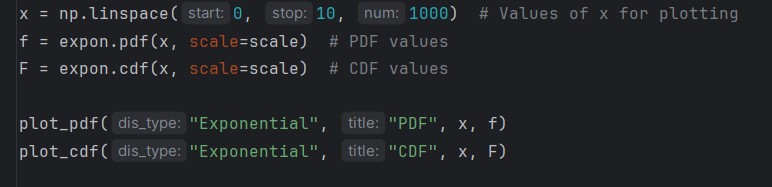
* + Calculating the Cumulative distribution function (CDF) at a given value using functions from the expon



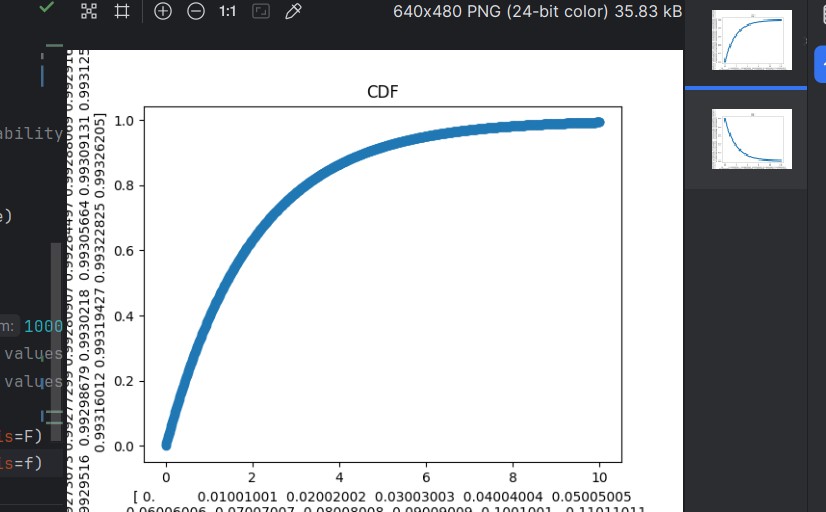
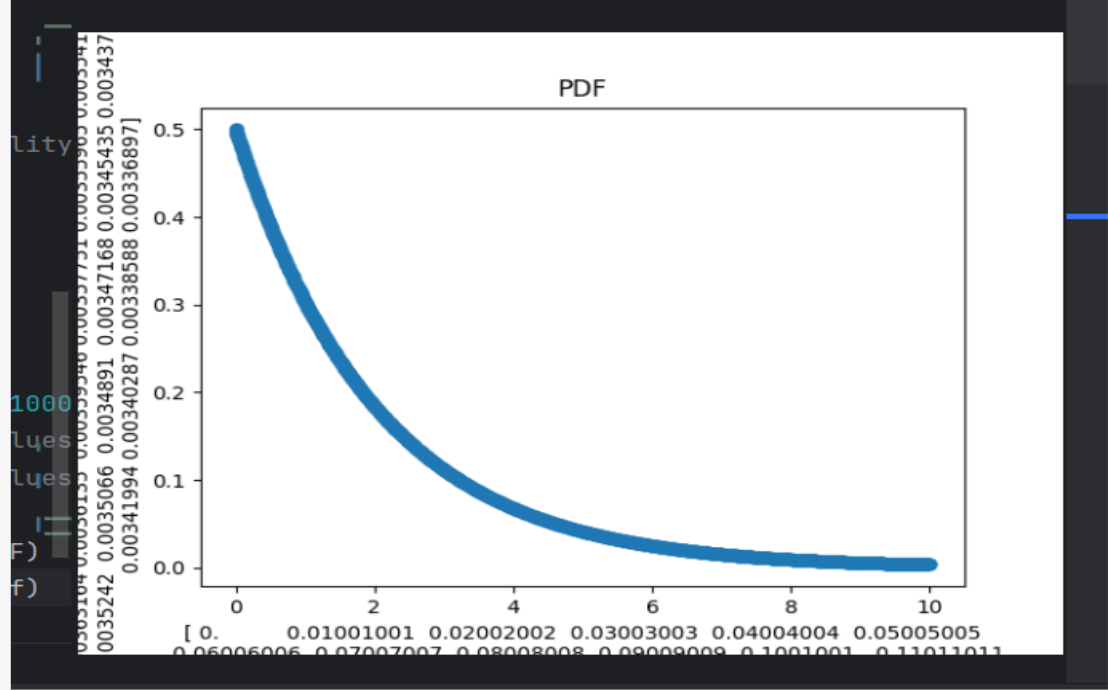
Output

The code generates plots of the probability density function (PDF) and cumulative distribution function (CDF) of the exponential distribution. It creates an array

of x values using np.linspace() 1000 values between 0 and 10 and calculates the corresponding PDF and CDF values using expon.pdf() and expon.cdf().

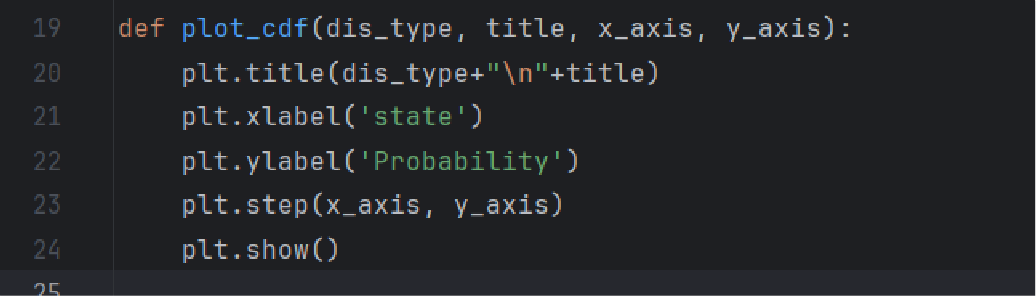




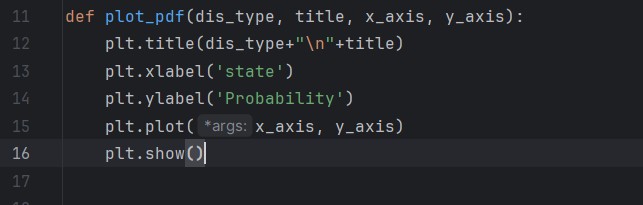




Function plot\_cdf :



Function plot\_pdf :



Definition:

Gaussian random variable

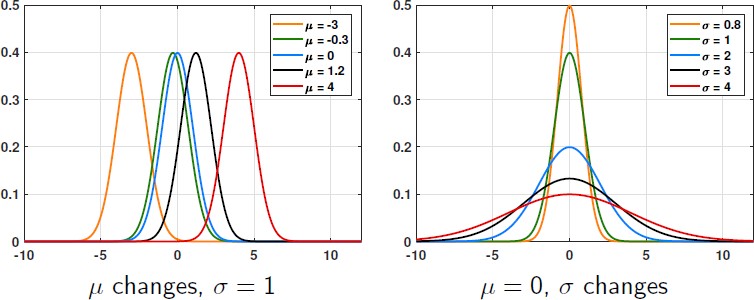
A continous random variable with probability density function (PDF)

of the form P(x) = 1 𝑒

√2𝜋𝜎2

−(𝑥−𝜇)2 2𝜎2

Where (𝜇, 𝜎2) are parameters of the distribution



The Cumulative distribution function:

CDF is denoted with the capital Greek letter phi Φ is the intgral

1 𝑥

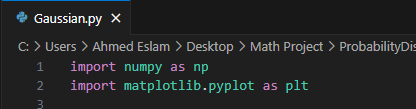
−𝑡2

Φ(x) = Fx(x) = √2𝜋 ∫−∞ 𝑒 2

𝑑𝑡

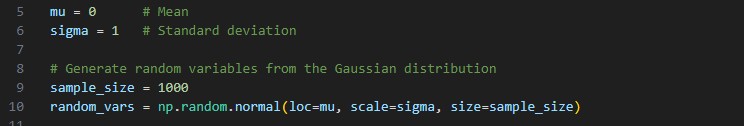
The Code explaination

1. Importing needed libraries



"numpy" as np for numerical operations, "matplotlib.pyplot" as plt for graphing the function

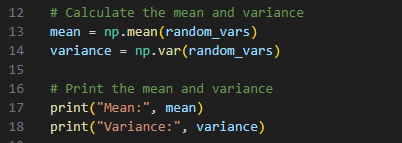
1. Setting parameters for Gaussian and generating random variables



The mean (mu) is the average value and standard deviation (sigma) is a measure of the spread of the distribution

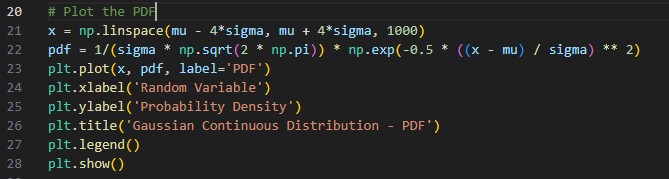
"np.random.normal" generates a set of random number following a normal distribution.

1. Calculating and printing Mean and Variance



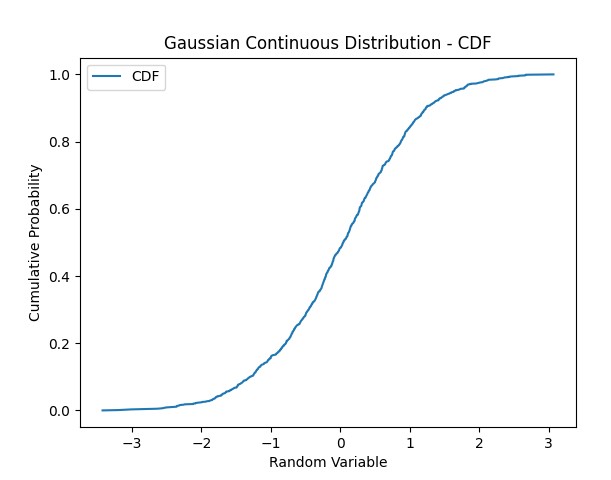
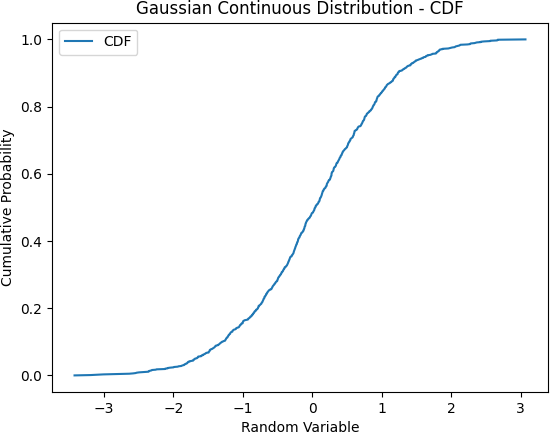
"np.mean","np.var" calculate mean and variance of the generated random variables from step 2

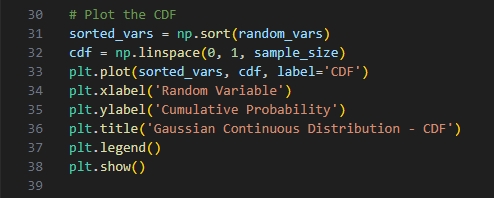
1. Plotting Probability Density Function (PDF)



"np.linespace" creates an array of evenly spaced values between two values PDF is calculated and plotted by "plt.plot" and displayed by "plt.show()"

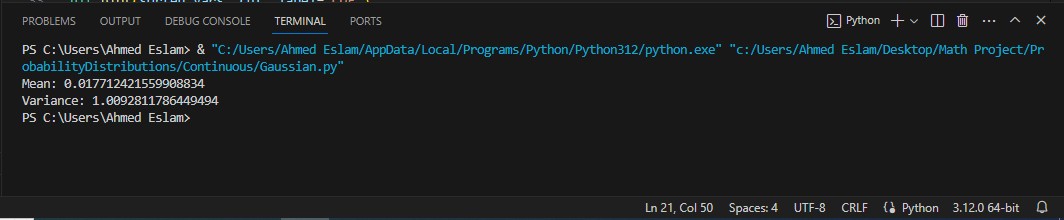
1. Plotting Cumulative Distribution Function





"np.sort" sort the generated random variable in ascending order Output:

1. Mean and Variance



1. PDF and CDF

