**1/1/2023**



**Random Variables**

**EMP305**

**Engineering Mathematics (5)**



**Random Variables & Python Project**

**Engineering Mathematics (5)**

**EMP305**

**Submitted by:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S# | Student Name | Edu Email | Student ID | Marks | | |
| Report & Slides  (30) | Implementation  )50) | Presentation  (20) |
| 1 | Ahmed Rafat | ahmed402812@feng.bu.edu.eg | 231903661 |  |  |  |
| 2 | Ahmed Eslam Foad | ahmed402455@feng.bu.edu.eg | 231903658 |  |
| 3 | Raghad Amr | raghad435083@feng.bu.edu.eg | 231903551 |  |
| 4 | Yasmine Yahia Elmetwally | yasmin402552@feng.bu.edu.eg | 221903092 |  |
| 5 | Mariam Mohamed | Maryam20889@feng.bu.edu.eg | 221903111 |  |
| 6 | Ahmed Sherif | ahmed442274@feng.bu.edu.eg | 231903592 |  |
| 7 | Mahmoud Mansour | mahmoud403355@feng.bu.edu.eg | 231903562 |  |

|  |
| --- |
| Supervised by:  Dr. Lamia Alrefaai |

1. **Introduction:**• Distribution function, mathematical expression that describes the probability that a system will take on a specific value or set of values, the classic examples are associated with games of chance.

• The random variables are used in many applications in our practical life, and through them we can determine many things, including determining the time of disasters and determining the validity period of many devices such as electronic devices.

1. **Types of Random Variable**

As discussed in the introduction, there are two random variables, such as:

* Discrete Random Variable
* Continuous Random Variable
* In our project we discuss some types of random variable for desecrate random variables
* we discuss Bernoulli, Binomial, Uniform, Geometric and Passion distribution.
* for continuous distribution we discuss Uniform, Exponential, and Gaussian distribution.
* And discuss some application and the importance of random variables in our real life.
* **Discrete Random Variable:**

A discrete random variable can take only a finite number of distinct values such as 0, 1, 2, 3, 4, … and so on. The probability distribution of a random variable has a list of probabilities compared with each of its possible values known as probability mass function.

1. **Bernoulli Distribution:**
2. **Definition**

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.

1. **Equation**:

PMF pX (0) = 1−p pX (1) =p

CDF F(x)=

1. **Properties:**

**X∼Bernoulli(p)**

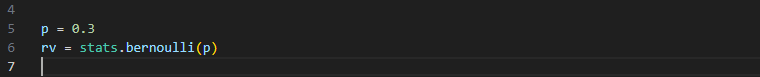
1. **Mean:** The expected value of a Bernoulli random variable X is **(**E [x] = P)
2. **E []** = P
3. **Variance:** Var [x] = P(1-P)
4. **Code**

#### First: we need to import needed libraries and define the plotting

* **"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

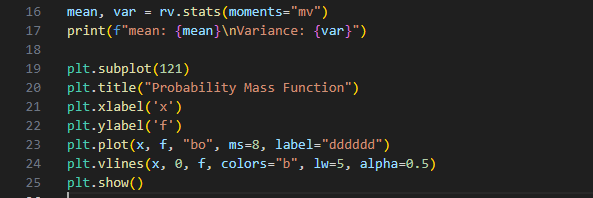
distribution with specified probability 'P'.

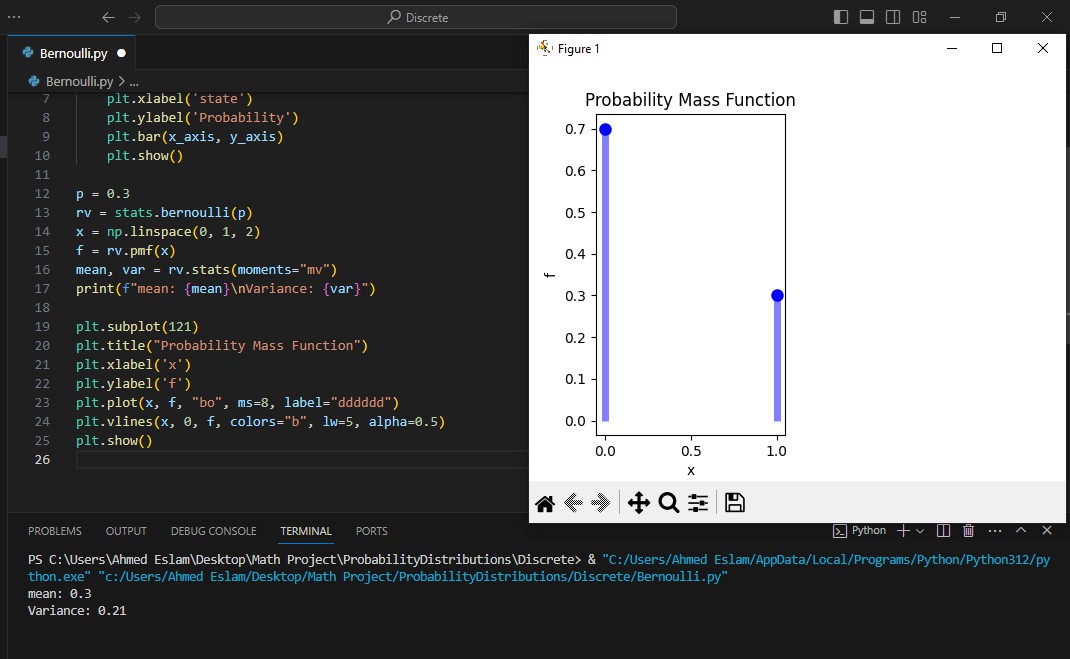
#### Third: Defining values for X (Possible outcomes):

* **'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].

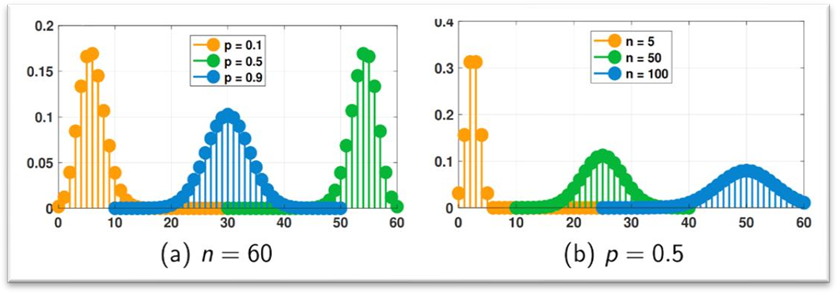
#### Fourth: calculating probability mass function (PFM), Mean and Variance:

* 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:

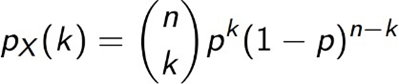


1. **Output**
2. **Binomial Distribution:**
3. **Definition:**

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."



1. **Equation:**

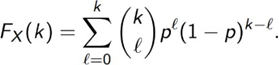
****

**PMF**

Where:

**n:** is number of trials

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



**CDF**

1. **properties:**

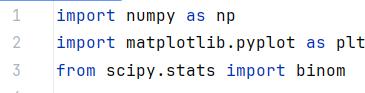
**X∼Binomial (n, p)**

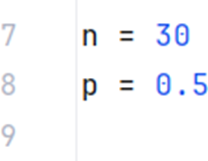
**1-Mean:** the expected value of X is( E[X] =np).

**2-E [] =** np (np+ (1−p)).

**3-Variance:** (Var[X] =np(1−p)).

1. **Code**

* **First:** 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

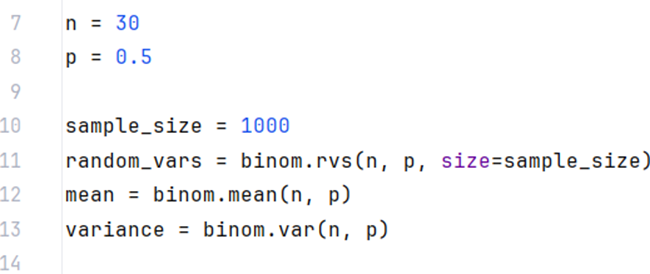


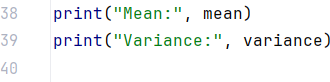
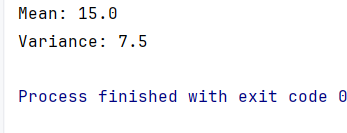
* N is number of trials while p is probability of success note that p should be a value between 0 and 1
* After that we generate the Mean, Variance and random variables

The **binom.rvs()** , **binom.mean()** and **binom.var()** functions are

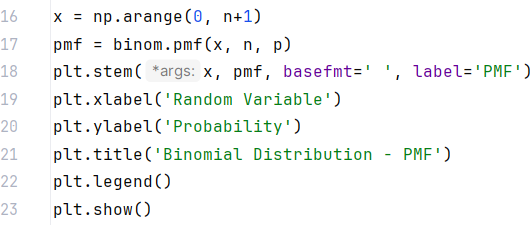
associated with the **scipy.stats library** and used for calculating mean and

variance and random variable using n and p ‘as in the context of the code’

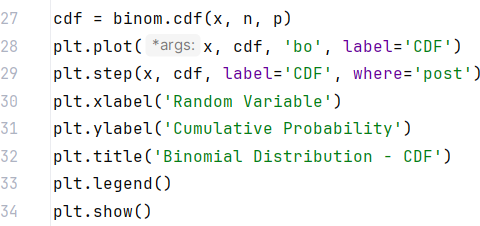


* We use the following code to print mean and variance:
* **Output**:
* After we will all that we will plot the PMF and CDF:

**PDF:**

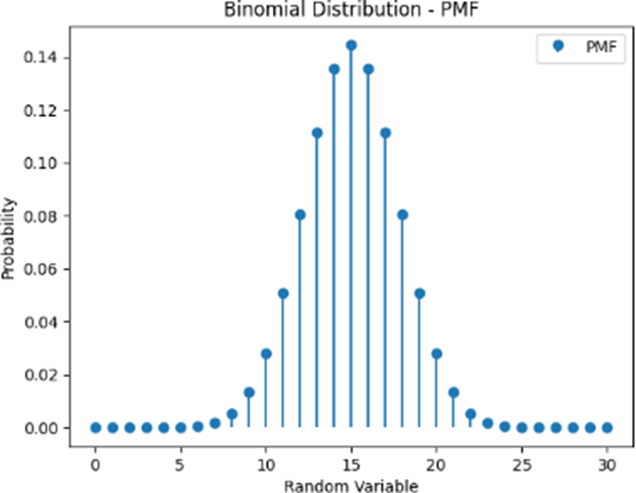


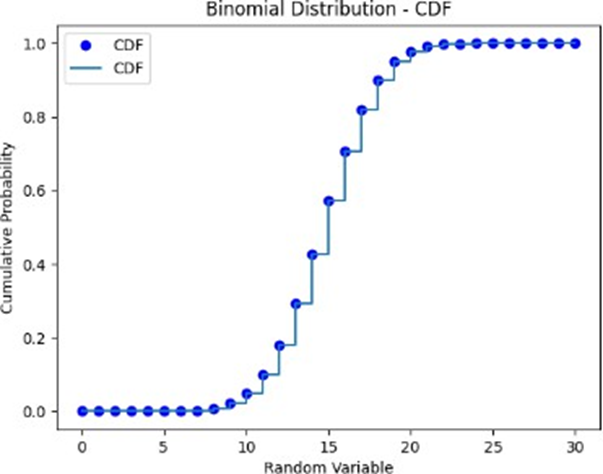
* For the previous code:
* **np.arange()** This function creates an array x using **numpy's** `arange` function. The array contains values from 0 to n
* **binom.pmf()** this function calculates the PMF values using X array and n and p which are previously initiated
* **plt.stem** function used to create 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 PMF and 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

1. **Results**





1. **Geometric Distribution:**
2. **Definition**

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.

1. **Equation**

**Px (k) = P (1 – P )k-1**

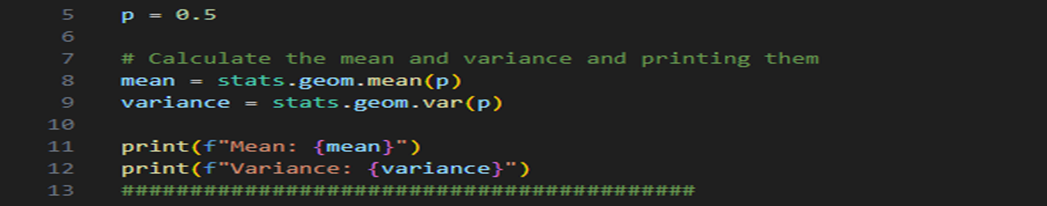
1. **properties:**

**X ~ Geometric (p)**

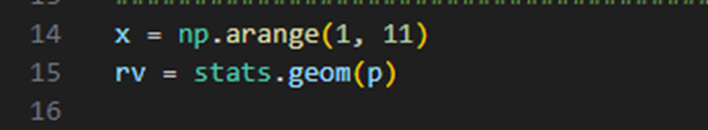
1. **Mean: 𝐸 ( 𝑥 ) =**
2. **𝐸 ( ):**-
3. **𝑉𝑎𝑟 ( 𝑥** )**:**
4. **Code**

#### First: Importing needed libiraries:

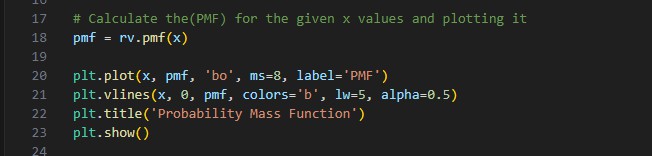
* **"numpy"** as **np** for numerical operations, **"scipy,stats"** as **stats** for statistical functions, **"matplotlib.pyplot"** as **plt** for graphing the functions.
* **Second:** Defining the probability and calculating Mean and Variance

****

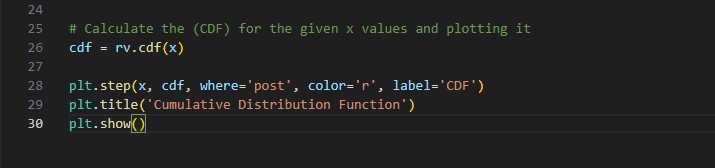
* Here "p" has 0.5 value that represent probability of success in geometric distribution Moreover, using "stats.geom.mean" and "stats.geom.var" from "scipy" libirary to calculate Mean and Variance for "p"
* **Third:** Creating values for x and Geometric Distribution

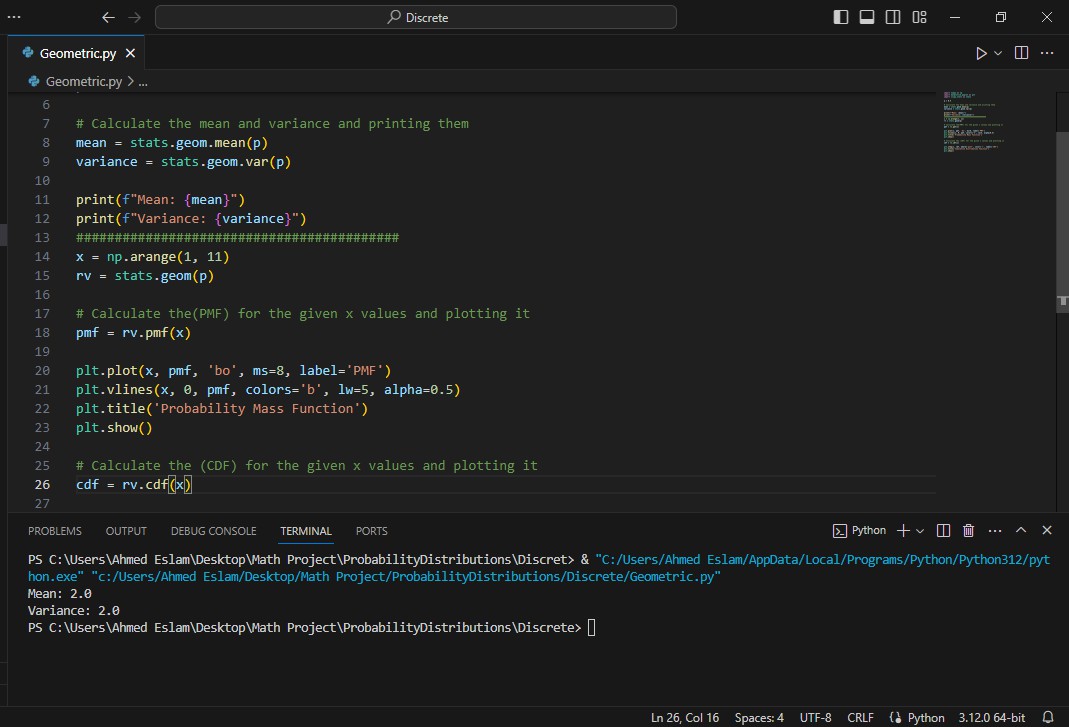
****

* 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

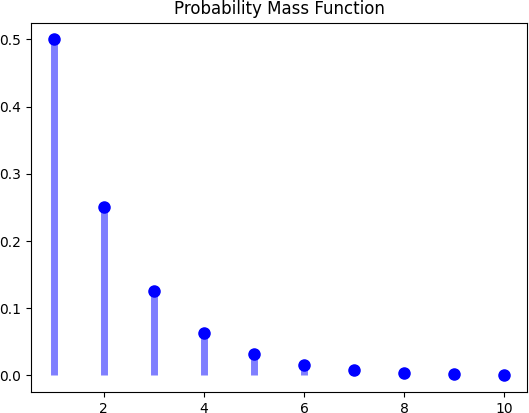


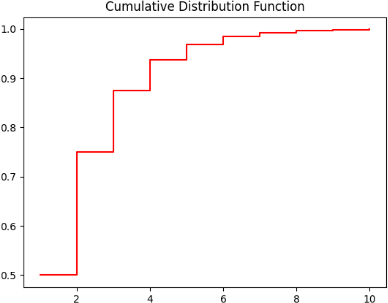
* **Fifth**: Calculating cumulative density function for "x" and plotting it

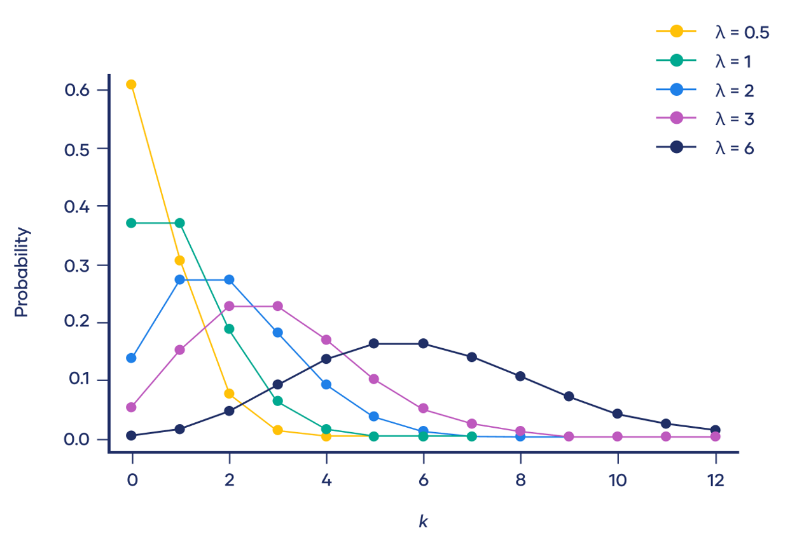


1. **Results:**

**Mean and Variance**





1. **Poisson Distribution:**
2. **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.
3. **Formula**

Poisson distribution formula is:

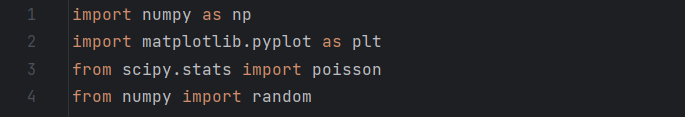
P(X = k) = \dfrac{e^{-\lambda} \lambda^k}{k!}

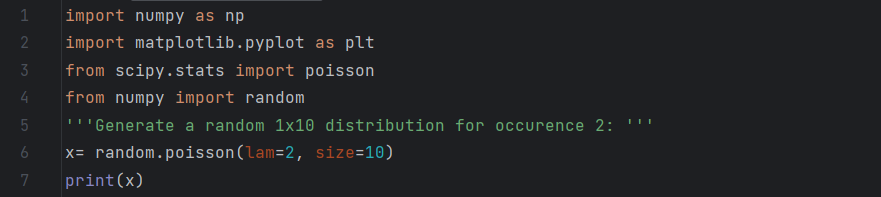
* Where:
* X is a random variable following a Poisson distribution
* k is the number of times an event occurs
* P (X = k) Is the probability that an event will occur k times
* e is Euler’s constant (approximately 2.718)
* \lambda is the average number of times an event occurs
* ! is the factorial function

1. **Properties of Poisson Distribution**

**X ~ Poisson ( 𝜆 )**

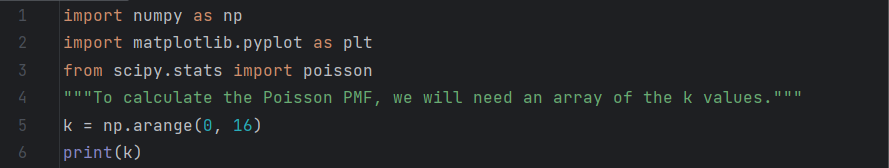
1. **Mean:** 𝐸(𝑥) = 𝜆
2. **𝐸 ()** = 𝜆 +
3. **variance**: 𝑉𝑎𝑟 (𝑥) = 𝜆
4. **Code**

* First, we start byimporting 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.

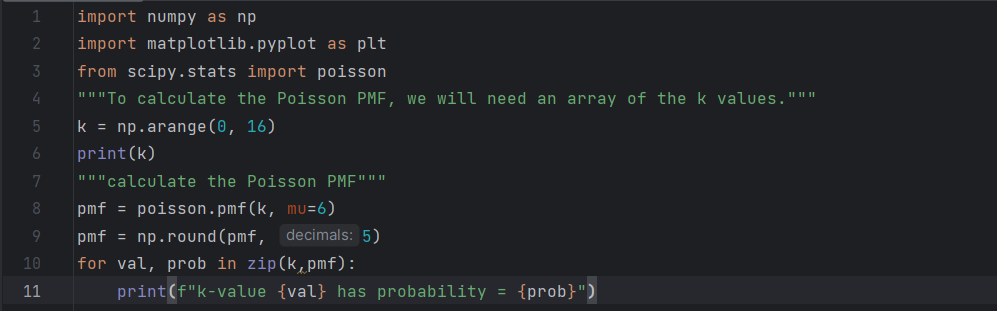


* Output:

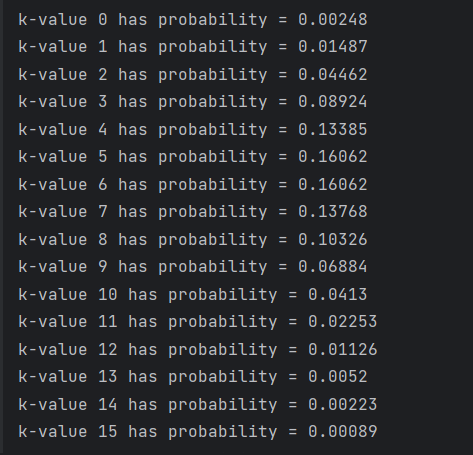
* Here, we need an array of the **k** values. We use **“np.arange ( )”** function and gave it numbers from 0 to 16 and print it.
* create an array with these values:



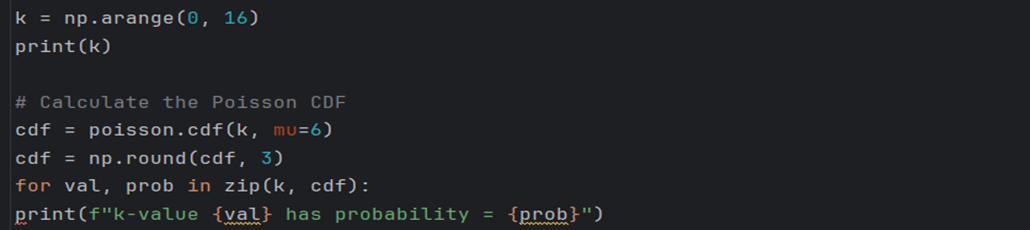
* Output:
* to calculate the Poisson PMF, we will use the **“poisson.pmf ( )”** method of the “**[Scipy.poisson](https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.poisson.html" \t "_blank)”** generator. It will need two parameters:
* k value (the **k** array that we created).
* μ value (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.



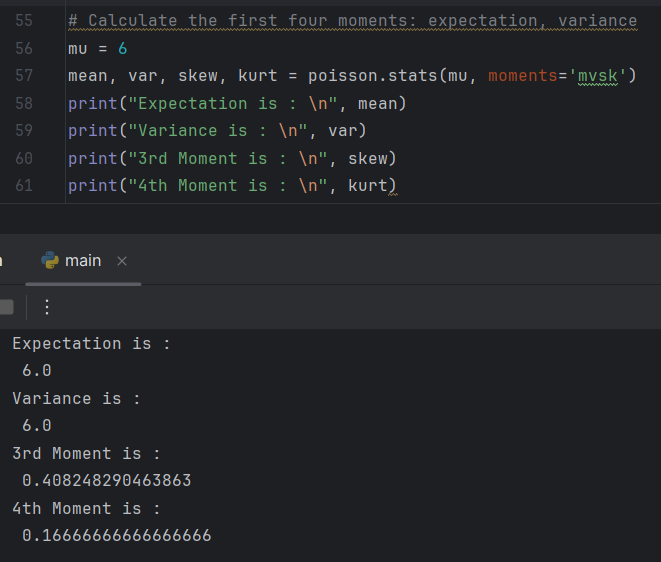
* **Output:**



* The same steps as PMF exactly except the function which is **“poisson.cdf ()”.**
* **“np.round()”:**This is a NumPy function that rounds each element to three decimal places.



* 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 and moments.

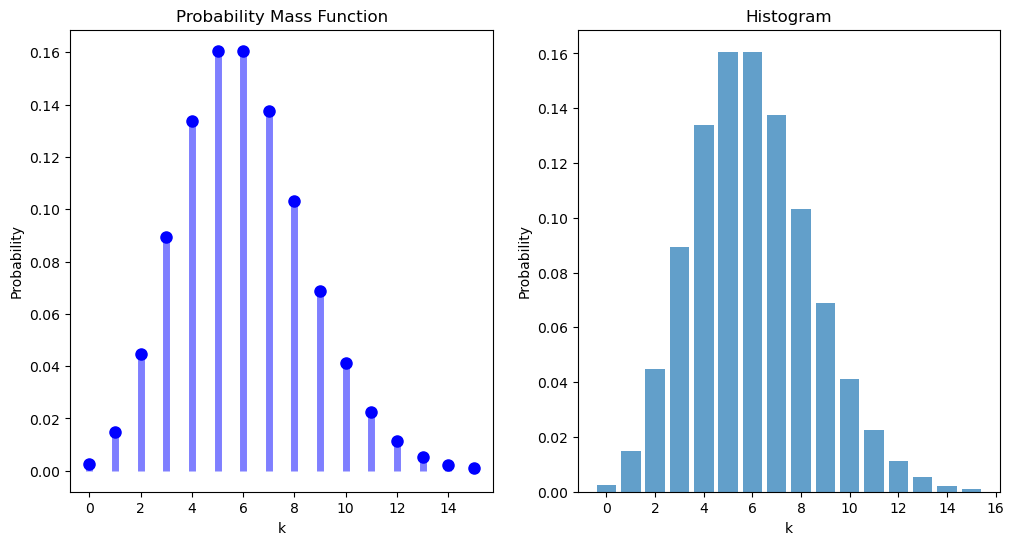


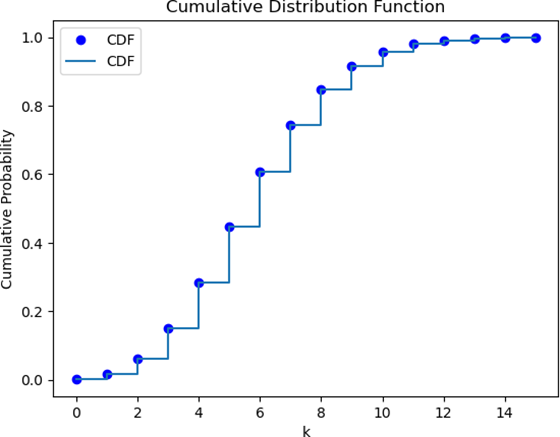
* We will need the **k** values array that we created earlier as well as the **PMF** values array in this step.
* By using **matplotlib** **library**.
* We need to plot PMF graph so, we use **“plt.bar( )”** function takes PMF value and k.
* Then, we add label to X-axis and Y-axis by using **“plt.xlabel ( )”** and **“plt.ylabel ( )”.**
* The same steps as PMF exactly except the function which is **“poisson.cdf ()”.**

****

1. **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.



 **(The PMF plot)**

**(The CDF plot)**

1. **Uniform Distribution:**
2. **Definition**

The discrete uniform distribution occurs when there are a finite number (m) of equally likely outcomes possible.

1. **Equation:**

PMF p(x) = 1 / m, where x=1,2,…,m

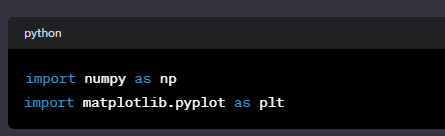
1. **Properties:**

**X∼Uniform(m)**

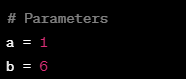
**1- Mean:** The expected value of a Uniform random variable X is (E[x]= (m+1) /2)

**2-** **Variance:** Var[x] = (m2 - 1) / 12

**IV. Code**



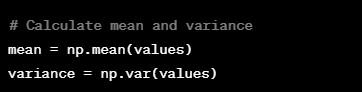
* This section imports the necessary libraries: **numpy** for numerical operations and **matplotlib.pyplot** for plotting.



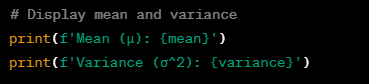
* Here, you define the parameters **a** and **b**, which represent the range of values for a discrete uniform distribution.



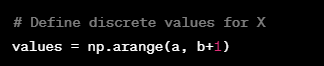
* This line generates 1000 random samples from a uniform distribution in the range **[a, b+1]** using **numpy.random.uniform**. These samples represent the random variable X.



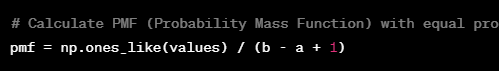
* Here, you calculate the mean and variance of the generated samples using **numpy.mean** and **numpy.var**.



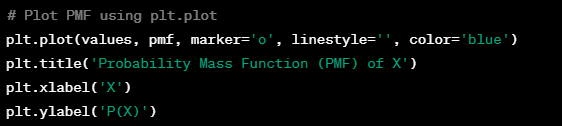
* This prints the calculated mean and variance to the console.



* This line creates an array of discrete values for X in the range **[a, b+1]** using **numpy.arange**.

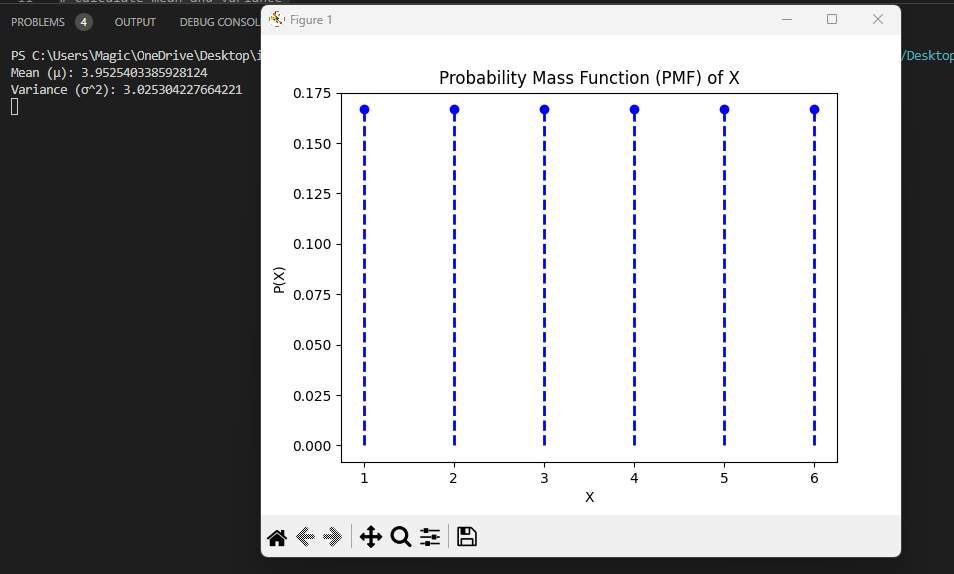


* Here, you calculate the Probability Mass Function (PMF) for the discrete values. In a discrete uniform distribution, each value has an equal probability, so the PMF is a constant value for each possible outcome.



* This block of code plots the PMF using **plt.plot**. The **marker='o'** option adds circular markers for each point.
* The **title**, **xlabel**, and **ylabel** functions add labels and a title to the plot.

**V. Results**

****

* **Continuous Random Variable**

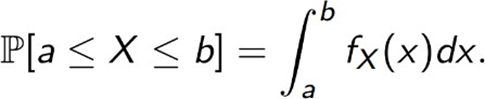
Continuous random variable is a random variable that has only continuous values.

Continuous values are uncountable and are related to real numbers

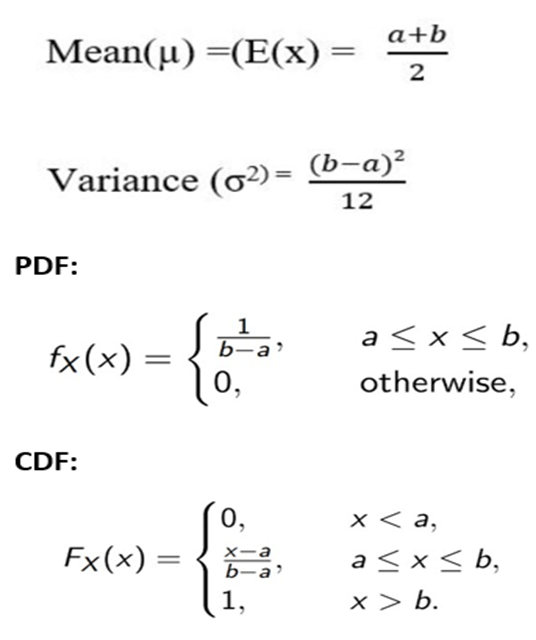
1. **Continuous uniform distribution:**
2. **Definition:**

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.

1. **Equation**

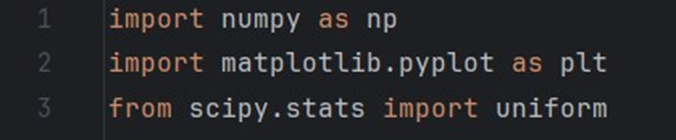


1. **Properties:**



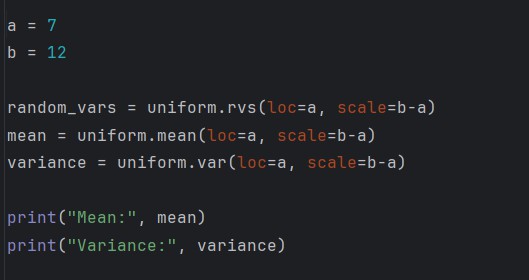
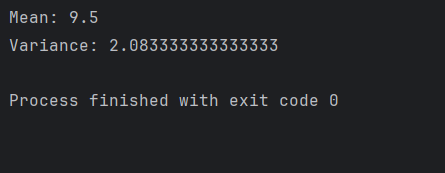
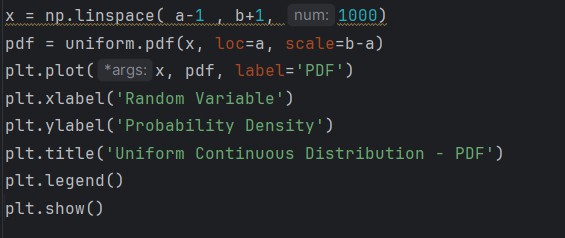
1. **Code**

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

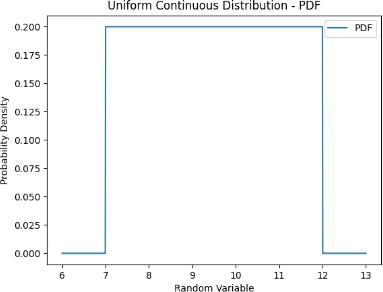


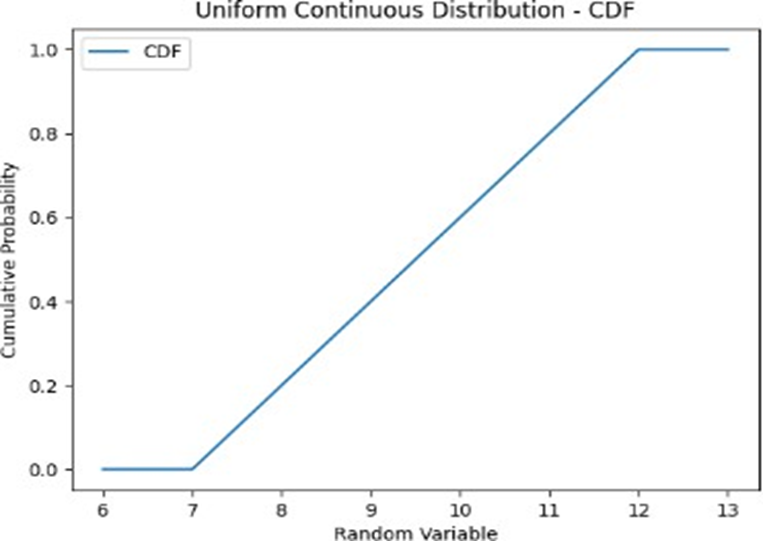
* **Line 1:** **numpy** library helps in mathematical and logical operations on arrays
* **Line 2:** we use **matplotlib library** for used for plotting the pdf and cdf
* **Line 3:** the **scipy.stats library** used for statistical and probabilistic operations.
* **Second:** To calculate and plot the 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
* **output** of printing Mean and Variance:
* **Third:** 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

1. **Results**



****

1. **Exponential Continuous Distribution**
   * 1. **Definition**

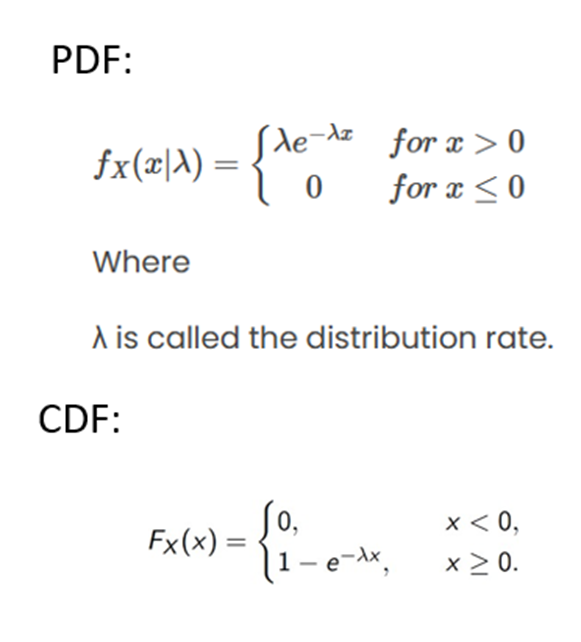
The Exponential distribution is a continuous probability distribution that often

concerns the amount of time until some specific event happens. It is a

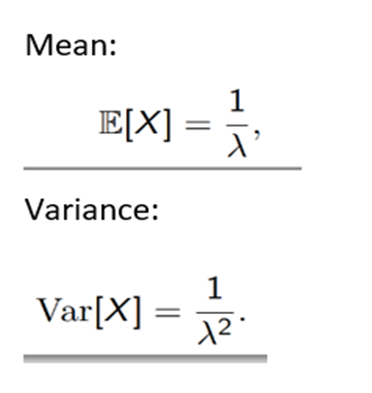
process in which events happen continuously and independently at a

constant average rate.

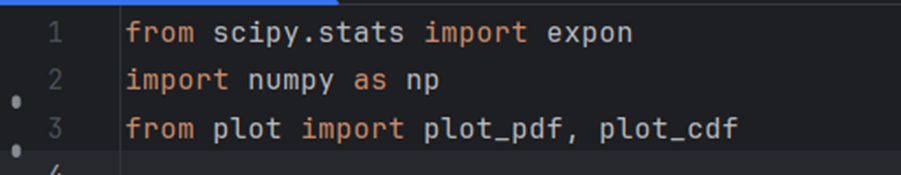
* + 1. **Equation**

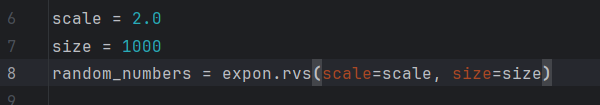
****

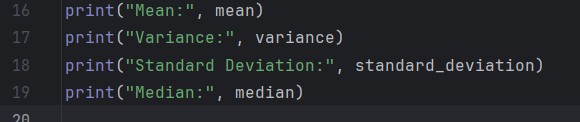
* + 1. **Properties:**

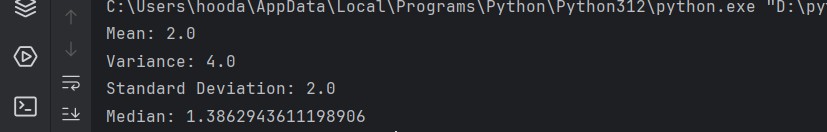
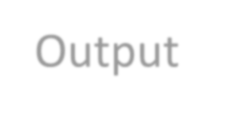
****

* + 1. **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

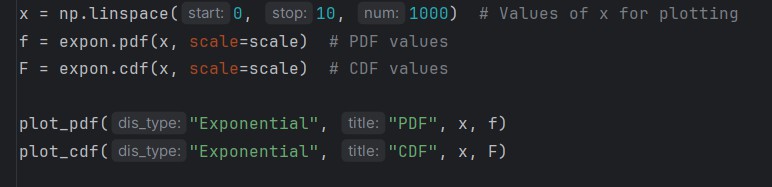
* Calculating the Cumulative distribution function (CDF) at a given value using functions from the expon



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

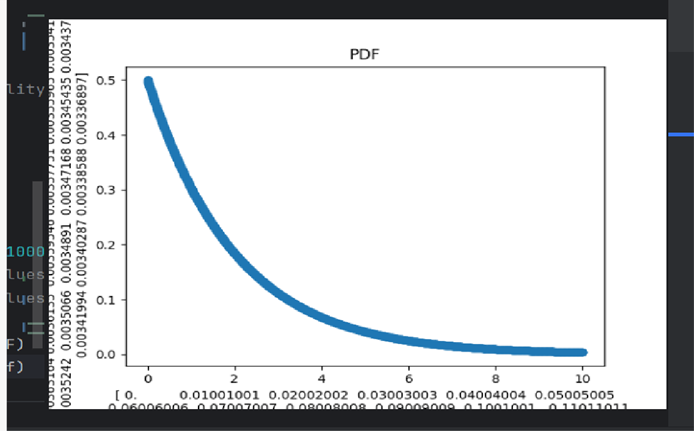
of x values using np.linspace() 1000 values between 0 and 10 and calculates the

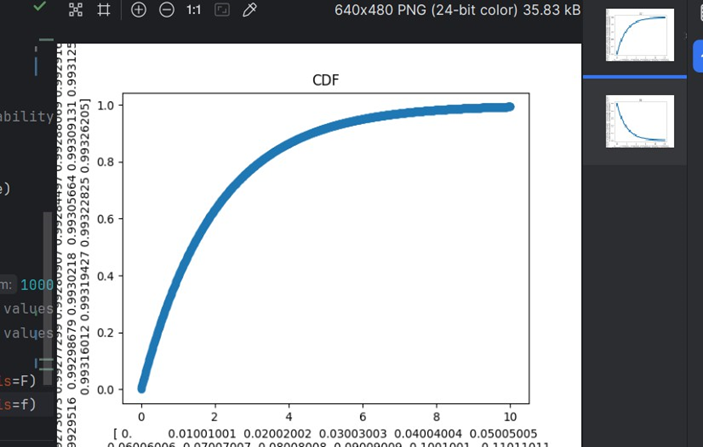
corresponding PDF and CDF values using expon.pdf() and expon.cdf().





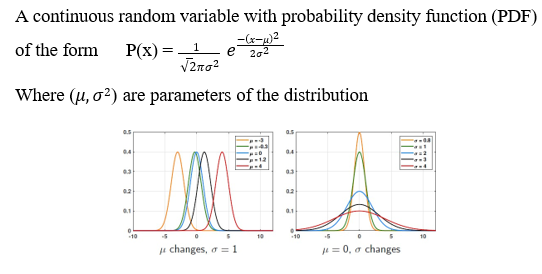
* + 1. **Results**

****

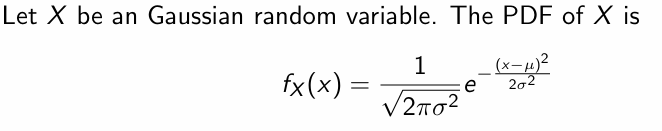
****

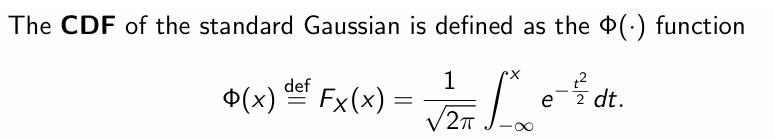
**3. Gaussian random variable**

1. **Definition:**

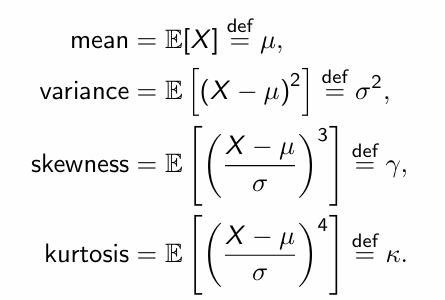
****

1. **Equation:**

****

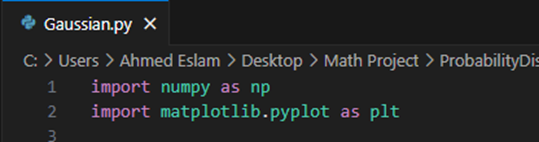


1. **Properties:**

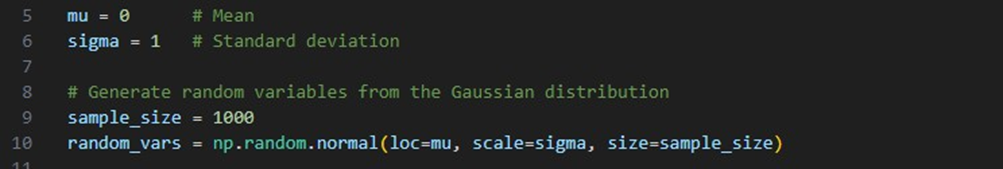
****

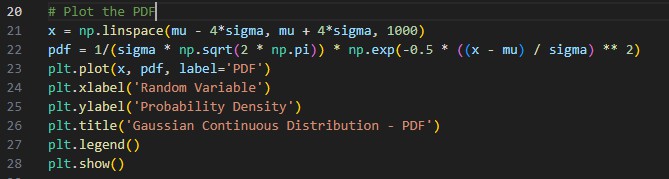
1. **Code**

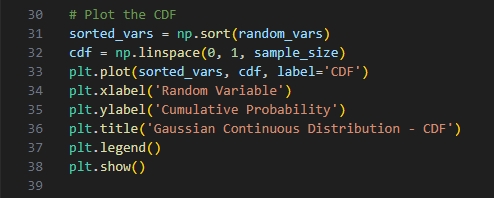
* **Frist** Importing needed libraries



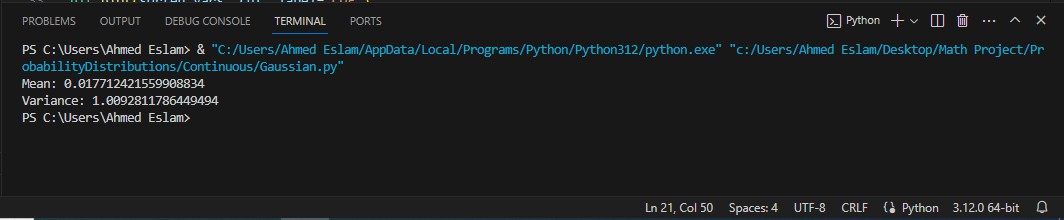
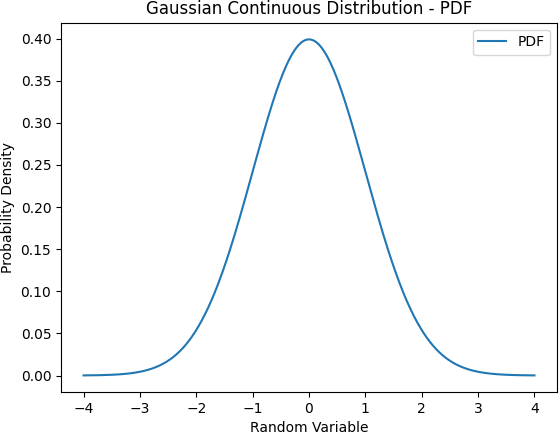
* "numpy" as np for numerical operations, "matplotlib.pyplot" as plt for graphing the function
* **Second** Setting parameters for Gaussian and generating random variables



* 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.
* **Third** Calculating and printing Mean and Variance
* **"np.mean","np.var"** calculate mean and variance of the generated random variables from step 2
* **Fourth** 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()"**
* **Fifth** Plotting Cumulative Distribution Function



* **"np.sort"** sort the generated random variable in ascending order

1. **Results:**
2. **Mean and Variance**
3. **PDF and CDF**

