Computational_Basics

January 29, 2024

1 Pset00

1.1 1.

1.1.1 a)

[1]:
```
primes = []
for i in range(2, 101):
    flag = True
    for j in range(2,i):
        if i % j == 0:
            flag = False
    if flag == True:
        primes.append(i)
print(primes)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73,
79, 83, 89, 97]
```

1.1.2 b)

[2]:
```
form_primes = []
for i in range(1,11):
    odd_num = (2**i) - 1
    for num in primes:
        if odd_num == num:
            form_primes.append(odd_num)
print(form_primes)
```

```
[3, 7, 31]
```

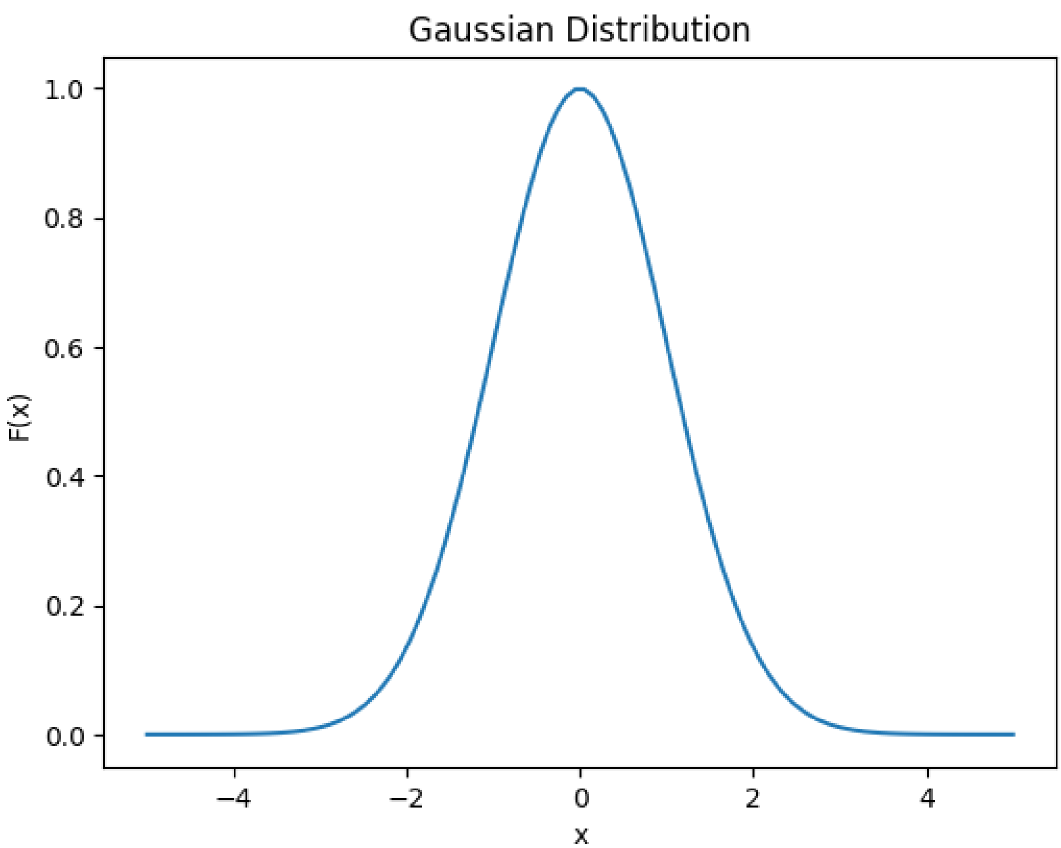
1.2 2.

1.2.1 a)

[3]:
```
import numpy as np
import matplotlib.pyplot as plt
c = 1
x_mean = 0
```

```
x_delta = 1
def F(x):
    return c*np.exp((-(x-x_mean)**2) / (2*(x_delta**2)))

x_vals = np.linspace(-5, 5, 100)
y_vals = F(x_vals)
plt.plot(x_vals, y_vals)
plt.title('Gaussian Distribution')
plt.xlabel('x')
plt.ylabel('F(x)')
plt.show()
```

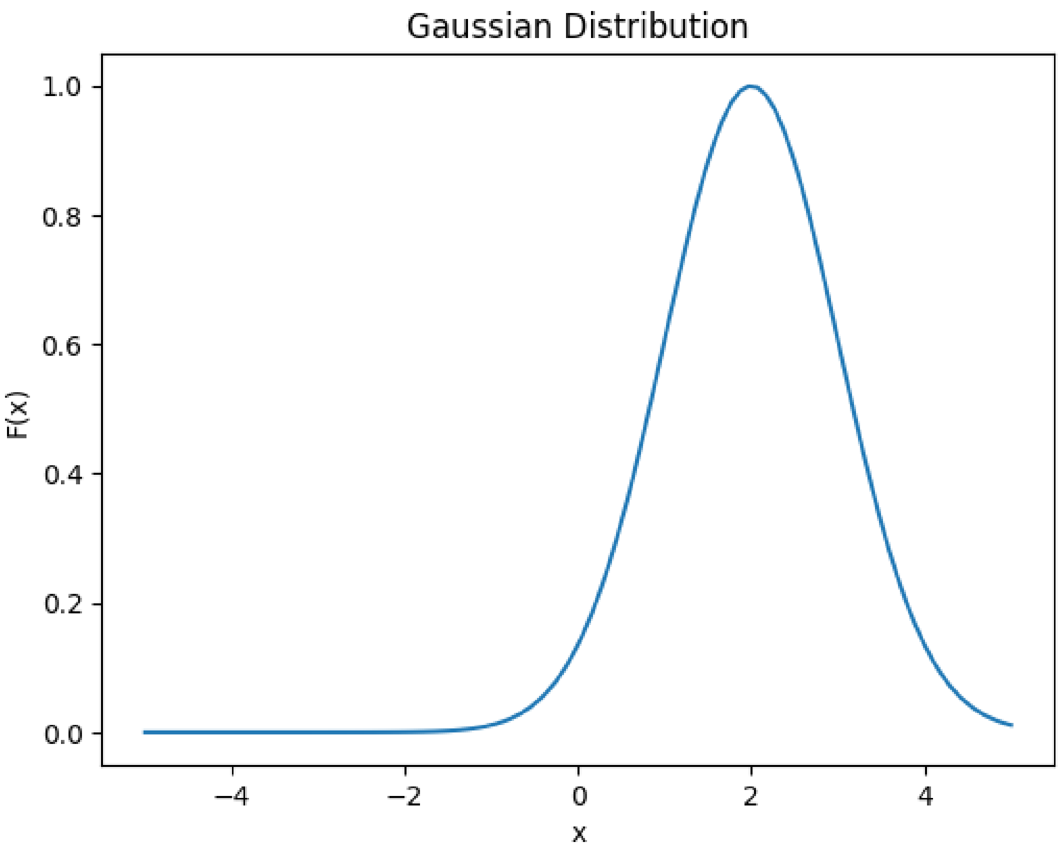


1.2.2 b)

Here I am changing the mean.

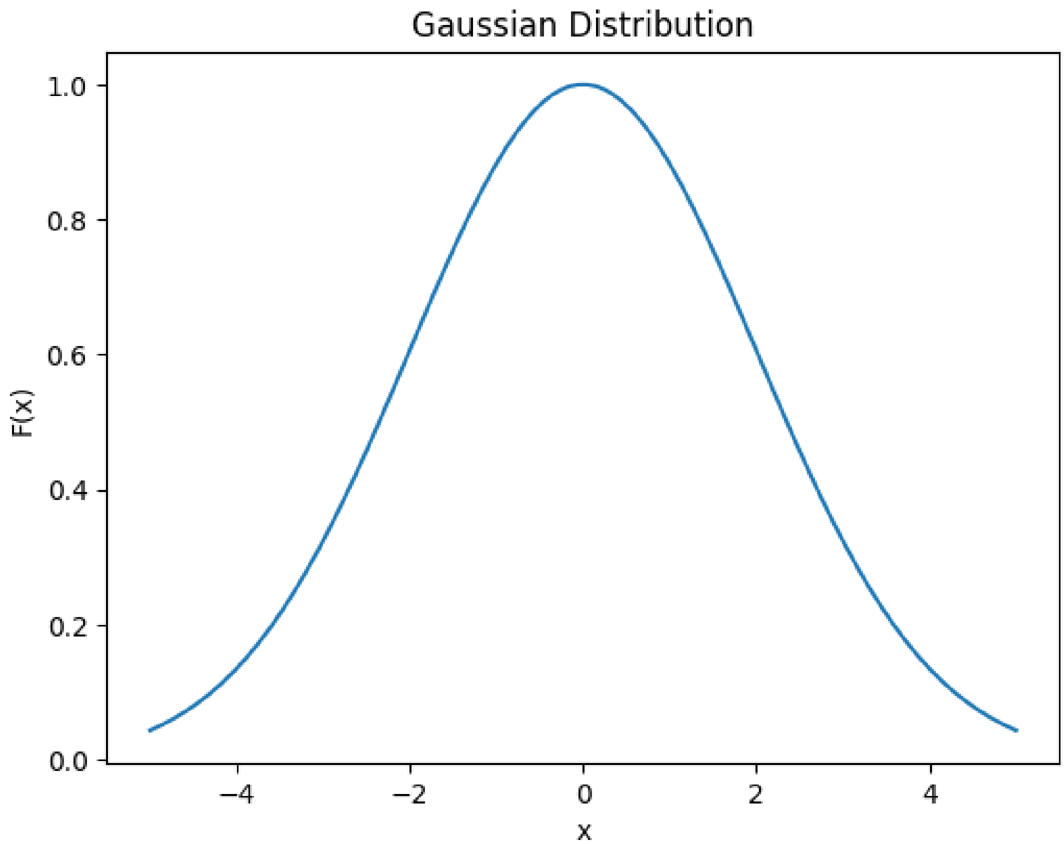
[4]:
```
x_mean = 2
x_vals = np.linspace(-5, 5, 100)
y_vals = F(x_vals)
plt.plot(x_vals, y_vals)
```

```
plt.title('Gaussian Distribution')
plt.xlabel('x')
plt.ylabel('F(x)')
plt.show()
```



Here I am changing delta x.

[5]:
```
x_mean = 0
x_delta = 2
x_vals = np.linspace(-5, 5, 100)
y_vals = F(x_vals)
plt.plot(x_vals, y_vals)
plt.title('Gaussian Distribution')
plt.xlabel('x')
plt.ylabel('F(x)')
plt.show()
print("The mean of x parameter causes the distribution to translate to the left␣
↪or right. Altering the delta of x causes the distribution to widen or narrow.
↪")
```



The mean of x parameter causes the distribution to translate to the left or right. Altering the delta of x causes the distribution to widen or narrow.

1.2.3 c)

#2

c)

$$\int_{-\infty}^{\infty} e^{-ax^2}\,dx = \sqrt{\frac{\pi}{a}}$$

$$a = \frac{1}{2\Delta x^2}$$

$$\downarrow$$

$$C\int_{-\infty}^{\infty} e^{-\frac{x^2}{2\Delta x^2}}\,dx = 1$$

$$\Downarrow$$

$$C = \Delta x\sqrt{2\pi} = 1$$

$$\boxed{C = \frac{1}{\Delta x\sqrt{2\pi}}}$$

1.2.4 d)

[6]:
```
from scipy import integrate

c = 1/(x_delta * np.sqrt(2 * np.pi))
x_mean = 0
x_delta = 1
lower_limit = -2.5 * x_delta
upper_limit =  2.5 * x_delta
result,error = integrate.quad(F, lower_limit, upper_limit)
print("Decimal percentage of distribution: ",result)
```

```
Decimal percentage of distribution:  0.49379033467422384
```

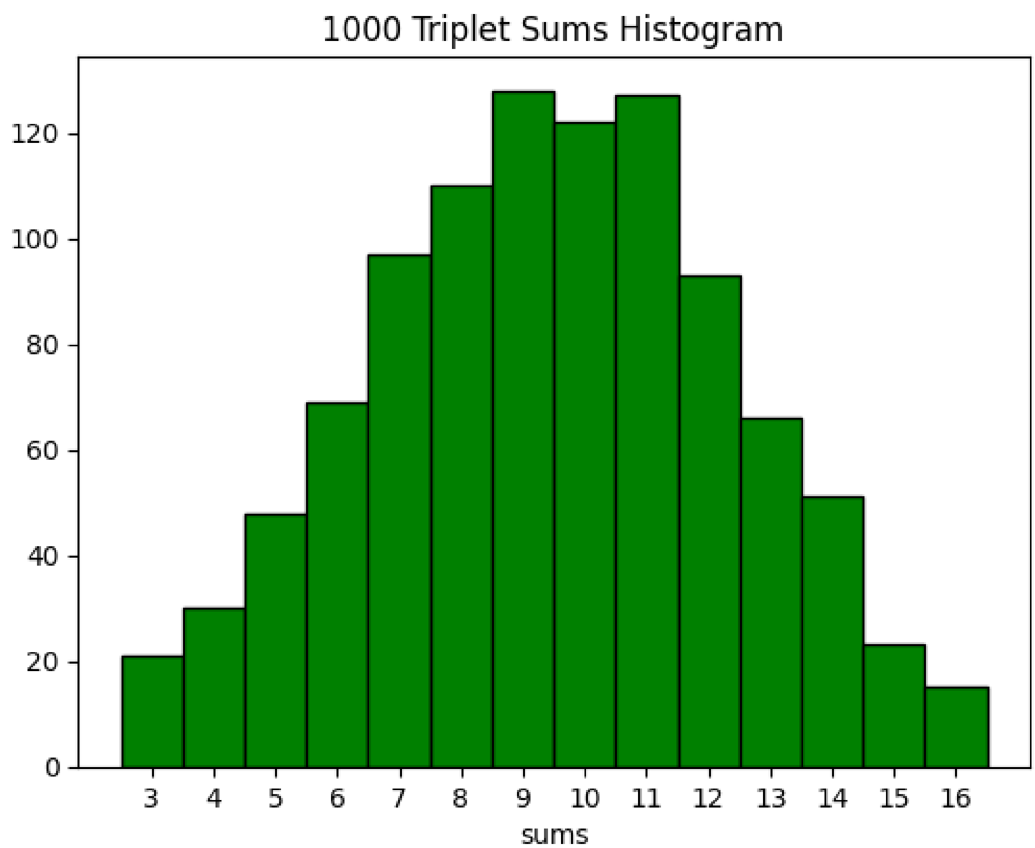
1.3 3.

1.3.1 a)

[7]:
```
import random as rand
import matplotlib.pyplot as plt
triplets_1k = []
for i in range(1000):
    triplet = []
    for j in range(3):
        triplet.append(rand.randint(1,6))
    triplets_1k.append(sum(triplet))

plt.title("1000 Triplet Sums Histogram")
n, bins, patches = plt.hist(triplets_1k, bins=14, color='green', edgecolor =␣
↪"black", align='left')
custom_xticks = bins[:-1]
custom_xlabels = ['3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13',␣
↪'14', '15', '16']  # Example custom tick labels
plt.xticks(custom_xticks, custom_xlabels)
plt.xlabel("sums")
```

[7]:
```
Text(0.5, 0, 'sums')
```

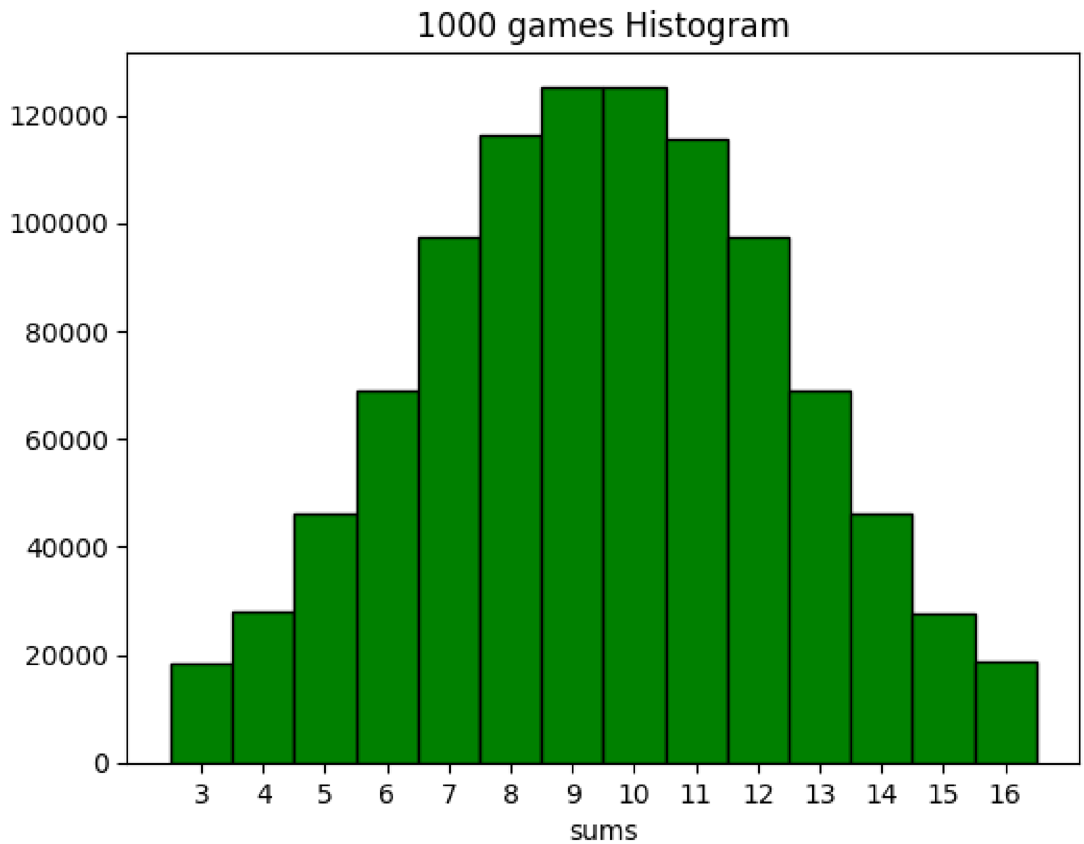


1.3.2 b)

```
[8]: games_1k = []
max_value = 0
for i in range(1000):
    triplets_1k = []
    for i in range(1000):
        triplet = []
        for j in range(3):
            triplet.append(rand.randint(1,6))
        triplets_1k.append(sum(triplet))
        if sum(triplet) == 18:
            max_value += 1
    games_1k += triplets_1k
plt.title("1000 games Histogram")
n, bins, patches = plt.hist(games_1k, bins=14, color='green', edgecolor =␣
↪"black", align='left')
custom_xticks = bins[:-1]
```

```
custom_xlabels = ['3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13',␣
↪'14', '15', '16']  # Example custom tick labels
plt.xticks(custom_xticks, custom_xlabels)
plt.xlabel("sums")
expected = (max_value/len(games_1k))*1000
print('The probability of rolling a triple-6 in a game: ', '{:.2f}'.
↪format(expected))
```

```
The probability of rolling a triple-6 in a game:  4.75
```



1.3.3 c)

[9]:
```
probability = 1/(6**3) #there are 6 options and rolling a 6 is one of them for␣
↪3 dice
ans = probability*1000
print('The number of times it is expected to roll 3 6\'s is: ', '{:.2f}'.
↪format(ans))
```

```
The number of times it is expected to roll 3 6's is:  4.63
```