

Hand-Made Central Limit Theorem

```
import matplotlib.pyplot as plt
import scipy.stats as sts
import math
%matplotlib inline
```

Central Limit Theorem says that if we have a large number of independent and identically distributed random variables, each with mean μ and dispersion σ^2 , and we select n i.i.d. random variables, their sum will be approximately distributed normally with mean $n\mu$ and dispersion $n\sigma^2$.

$$\bar{X}_n \approx N(\mathbb{E}\mu, \sigma^2/n)$$

Let us check if it really so for the case when $F(X)$ is a uniform distribution:

Define uniform distribution from a to b:

```
uniform_rv.rvs(10)
```

```
array([2.52015047, 4.91672774,
       2.41479864, 4.92748426,
```

In [112...

```
x = np.linspace(0,6,1000)
pdf = uniform_rv.pdf(x)
plt.plot(x, pdf)

plt.ylabel('$f(x)$')
plt.xlabel('$x$')

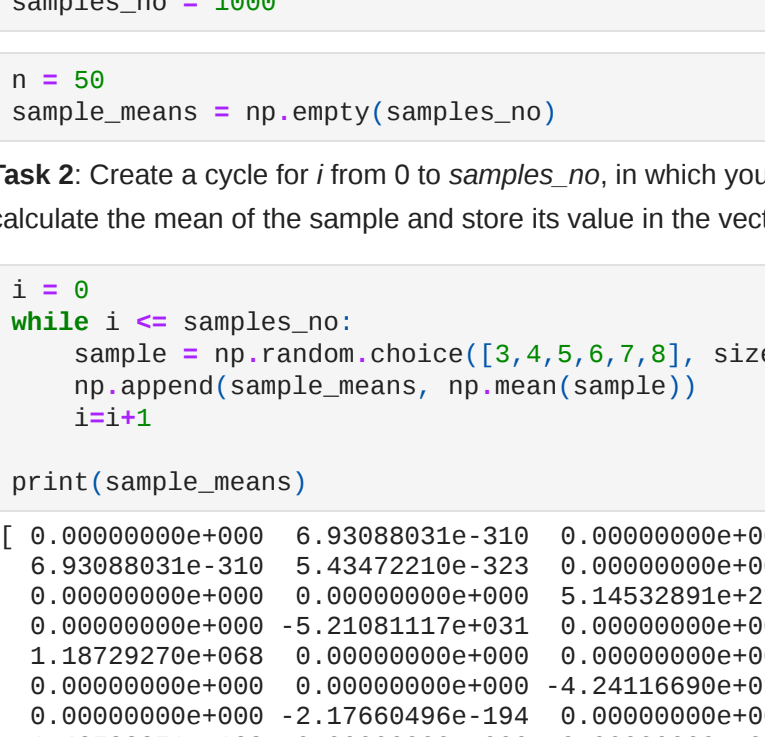
Text(0.5, 0, '$x$')
```

0.20				
------	--	--	--	--

0.15

Response	Proportion
No	0.01
Yes	0.09

each sample and store the



0.000000000e+000	0.000000000e+000	-5.18805516e-307	0.000000000e+000
0.000000000e+000	9.05664082e-006	0.000000000e+000	0.000000000e+000
-1.05901774e+080	0.000000000e+000	0.000000000e+000	-1.06812671e-156
0.000000000e+000	0.000000000e+000	-2.15091052e+022	0.000000000e+000
0.000000000e+000	1.44494218e+001	0.000000000e+000	0.000000000e+000

```
0.000000000e+000
0.000000000e+000
```

```

6.930880000e-310  6.93088017e-310
6.93088017e-310  4.60347913e-252

```

6.	930880832e-310	6.	930880817e-310	-3.69996998e-139	6.	93088032e-310	
-1.	930880817e-310	1.	59639856e+033	6.	93088027e-310	6.	41915047e-314
-6.	0.01813793e+154	6.	930880301e-310	6.	93088017e-310	-3.	97218764e+092
6.	930880831e-310	6.	930880817e-310	-7.	15946385e+126	6.	93088022e-310
6.	930880817e-310	1.	35996116e+069	6.	93088012e-310	6.	93088017e-310
6.	61396197e-031	6.	930880807e-310	6.	93088017e-310	9.	9056552e+111
6.	93087985e-310	6.	93088017e-310	2.	57949713e+254	6.	93088027e-310
6.	93088017e-310	5.	85416743e-245	6.	93088027e-310	6.	93088017e-310
7.	20757425e+205	6.	930880827e-310	6.	93088017e-310	-7.	56349865e+207
6.	930880808e-310	6.	93088050e-160	-7.	31729360e-005	6.	93088080e-310
6.	930880832e-310	1.	17066590e-160	6.	93088012e-310	6.	93088032e-310
-4.	65600496e+215	6.	930880827e-310	6.	93088032e-310	2.	0769459e-039
6.	93087993e-310	6.	930880832e-310	3.	40237951e+067	6.	93087993e-310
6.	93088032e-310	-1.	84834305e-193	6.	93087993e-310	6.	93088032e-310
9.	35892531e-263	6.	93088032e-310	6.	93088032e-310	6.	27277045e+218
6.	930880831e-310	6.	93088017e-310	1.	10559546e-258	6.	93088031e-310
6.	93088027e-310	1.	37433242e+021	6.	93088027e-310	6.	93088027e-310

6.93088032e-310	6.93088027e-310	-4.38657209e+229	6.93088013e-310
6.93088027e-310	-2.67999895e+223	6.93088013e-310	6.93088027e-310

6.93088012e-310 6.93088032e-310 -1.23766256e+051 6.93088032e-310
6.93088032e-310 6.62469832e+204 6.93088032e-310 6.93088032e-310

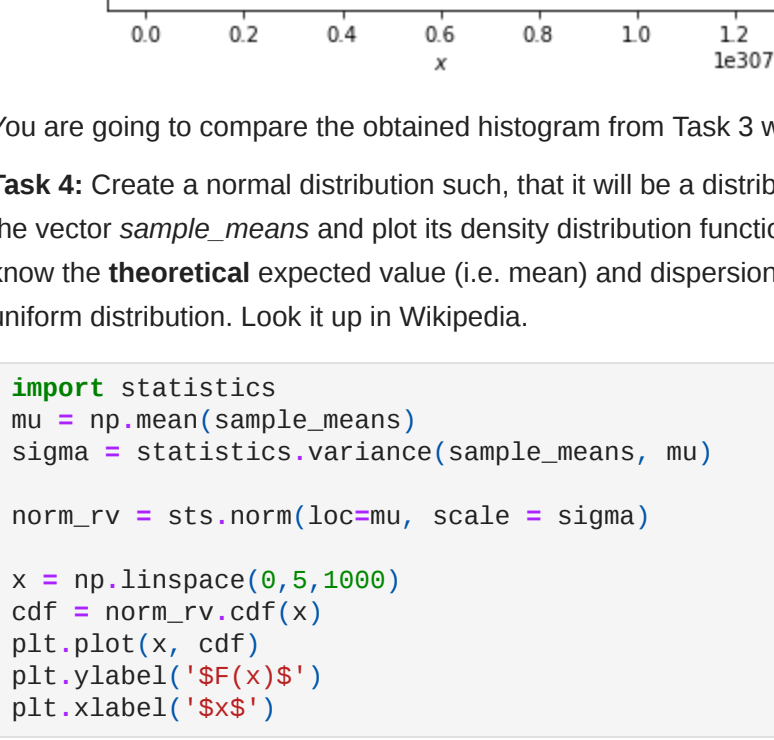
6.93088032e-310	6.93088032e-310	-5.83793389e+203	6
6.93088032e-310	9.78262830e+304	6.93088032e-310	6
-3.31143805e+046	6.93088022e-310	6.93088032e-310	1
6.93088017e-310	6.93086265e-310	-2.53894422e+287	6
6.93088027e-310	-1.69398497e-258	6.93088017e-310	6
-1.43339886e-069	6.93088017e-310	6.93088027e-310	-2

[illegible]

0.04

Fraction of Samples

-0.04



```
/opt/anaconda3/lib/python3.8/statistics.py:686: RuntimeWarning: overflow encountered in double_scalars
  T_total_count = sum((x-c)**2 for x in data)
```

Text(0.5, 0, '\$x\$')

The plot shows a single data point at the coordinates (0.5, 0). The point is represented by a small black dot. A label '\$x\$' is placed to the right of the point, indicating its x-value. The x-axis is labeled from 0 to 1, and the y-axis is labeled from -0.5 to 0.5.

Task 5 Plot the histogram from Task 3 and the distribution

```
plt.plot(x, cdf)
```

```
plt.hist(sample_means, density = True)
plt.ylabel('$F(x)$')
plt.xlabel('$x$')
```

```
#you should obtain something like this:
```

0.5-

