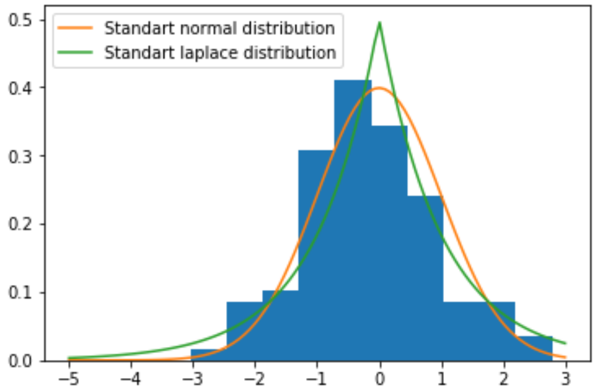
In [46]:
```
import numpy as np
import pandas as pd
import math
import matplotlib.pyplot as plt
import scipy
from scipy.stats import norm
from scipy.stats import expon
from scipy.stats import laplace as lapl
from statsmodels.distributions.empirical_distribution import ECDF
%matplotlib inline
```

In [47]:
```
X = np.load('9_3.npy')
print(len(X))
plt.hist(X, density=True)
grid = np.linspace(-5, 3, 100)
plt.plot(grid, norm.pdf(grid), label="Standart normal distribution")
plt.plot(grid, lapl.pdf(grid), label="Standart laplace distribution")
plt.legend()
plt.show()
np.array(X)
```

```
100
```



Out[47]:
```
array([ 1.21643369, -0.5583868 , -0.15235402,  0.04667652,  0.03756226,
       -1.0307989 , -1.8016716 , -0.95322444,  0.32009457, -0.4558245 ,
       -1.97227822,  0.16301045,  1.06237559, -1.43118931,  0.65767633,
        0.25289647,  1.63405579, -1.04036922, -0.44059907, -0.54180682,
       -0.43038967,  0.28904665, -3.0448604 ,  0.98288288,  0.87588642,
        0.33519881, -0.34514396, -1.57854064, -2.38437115,  0.60582575,
        1.01303185, -0.5836204 , -1.0168949 , -1.81102131, -0.85752717,
        1.21617105, -0.72473406, -1.06452162, -0.94196431, -0.94408012,
       -0.78465005,  0.83175143,  0.81165755, -0.88815803,  0.43270904,
       -1.10118264,  2.17239221,  1.61177944,  2.78710915, -1.24059136,
       -0.24864645,  0.42923075,  0.20269564, -0.57640622, -0.68811663,
       -0.73667925,  0.29039343, -0.48373756,  0.33121016, -0.17799872,
       -1.72426925, -0.12107439, -0.31743564,  0.38054239,  1.41347815,
       -0.42038784,  0.8271258 ,  0.9726375 , -0.08365299, -2.35752404,
       -0.97668201,  0.22428592, -1.92784232, -0.21298906, -0.36051004,
       -0.48381459,  0.19336207,  0.1006805 , -1.81247789,  2.01765167,
        0.99605834, -0.67942747, -0.34714667,  0.57461997,  0.12178738,
       -0.31228888, -0.68572104,  0.85553491, -0.56461697, -0.26920357,
        0.39870967, -0.74894349, -0.88944858,  1.9662598 ,  0.96880833,
       -2.18866493,  1.73847858,  0.84379246,  2.31285313, -0.94227081])
```

В качестве априорного распределения для σ, θ возьмём стандартное экспоненциальное Exp(1).
$q(t) = e^{-t} \cdot I[t \geq 0]$.
Байесовский критерий: $K = \frac{\int f_0(X,\sigma)q(\sigma)d\sigma}{\int f_1(X,\theta)\tilde{q}(\theta)d\theta}$, где f_0, f_1 - функции правдоподобия, соответствующие H_0, H_1. Т.е.

$$K = \frac{\int f_0(X,\sigma)q(\sigma)d\sigma}{\int f_1(X,\theta)\tilde{q}(\theta)d\theta} = \frac{\int_{-\infty}^{\infty}\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n e^{-\frac{\sum_{i=1}^n x_i^2}{2\sigma^2}} e^{-\sigma} I(\sigma > 0)d\sigma}{\int_{-\infty}^{\infty}\left(\frac{1}{2\theta}\right)^n e^{-\frac{\sum_{i=1}^n |X_i|}{\theta}} e^{-\theta} I(\theta > 0)d\theta} = \frac{\int_0^{\infty}\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n e^{-\frac{\sum_{i=1}^n x_i^2}{2\sigma^2}} e^{-\sigma}d\sigma}{\int_0^{\infty}\left(\frac{1}{2\theta}\right)^n e^{-\frac{\sum_{i=1}^n |X_i|}{\theta}} e^{-\theta}d\theta}.$$

In [48]:
```
N=10000
```

Сгенерируем выборку параметров размера N=10000 из стандартного экспоненциальног распределения. Также найдем подынтегральные функции в К.

In [49]:
```
t = expon.rvs(1,size = N)
def f0(sig, Y):
    return (((1/(2*np.pi*(sig**2)))**(len(Y)/2))*np.exp(-np.sum(Y**2)/(2*(si
g**2)))*np.exp(-sig))
def f1(th, Y):
    return ((1/(2*th))**len(Y)*np.exp(-np.sum(np.abs(Y))/th)*np.exp(-th))
```

Вычислим статистику К, сгенерировав выборку из нормального распределения $N(0, \sigma)$ для $\sigma \in t$.

In [50]:
```
n = 100
K = []
for sigma in t:
    Y = norm.rvs(scale=sigma**0.5, size = n)
    a = scipy.integrate.quad(f0, 0, +np.inf, args=(Y,))[0]
    b =  scipy.integrate.quad(f1, 0, +np.inf, args=(Y,))[0]
    K.append(a/b)
```

In [61]:
```
print(K[:20])
```

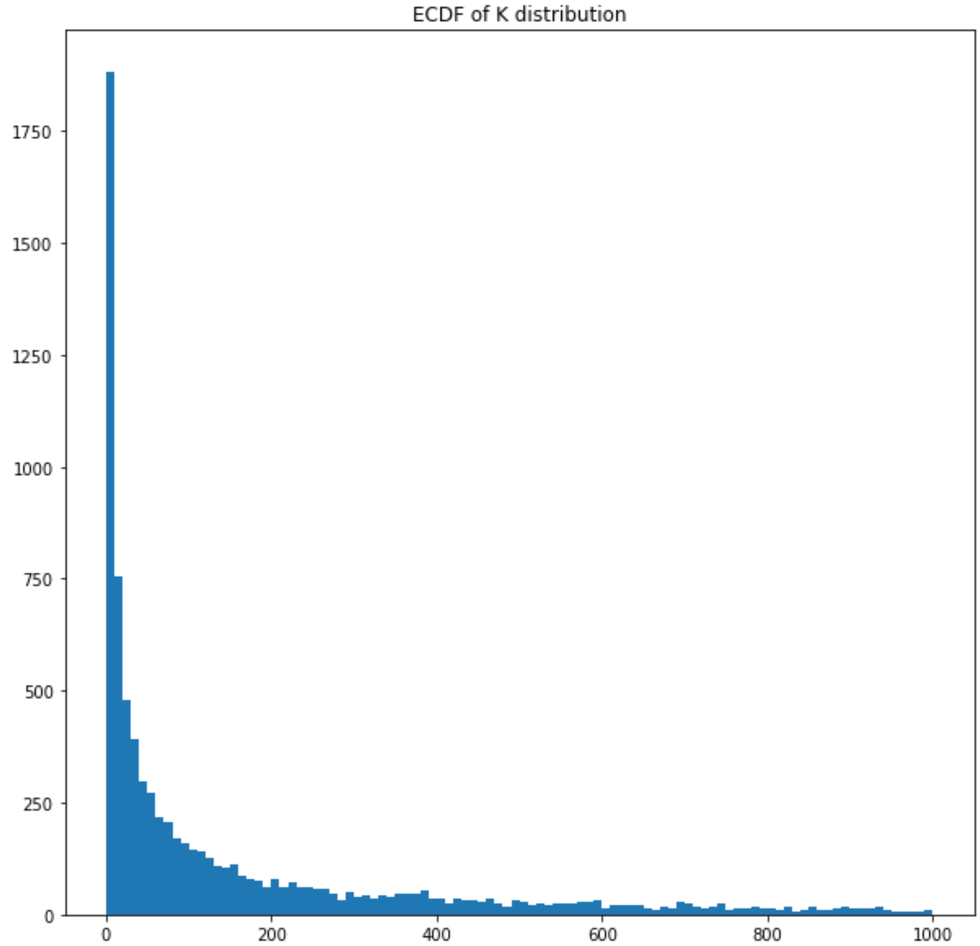
```
[0.0003052684411785413, 0.0006194700553622568, 0.0012120847857943584, 0.00274
72040693917155, 0.004699499615103332, 0.004837467432455402, 0.006077061169045
922, 0.00721210526876659, 0.00801153376327183, 0.008331247718289692, 0.009334
30362390315, 0.009951690175688047, 0.010598427351575682, 0.011641723873629148
, 0.018613159425992832, 0.01970936641762426, 0.020037515862999646, 0.02041954
047506885, 0.020936947716831182, 0.021333664117614365]
```

In [62]:
```
print(np.max(K))
print(np.min(K))
K.sort()
```

```
1478944.889174012
0.0003052684411785413
```

Нужно явным образом получить критерий $S = \{K \leq z_\alpha\}$. Построим эмпирическое распределение статистики K и найдем такое $z_{0.05}$, что $P(X \in S) = P(K \leqslant z_{0.05}) = \alpha = 0.05$ при условии, что H_0 верна(вероятность отвергнуть правильную гипотезу). Необходимо рассмотреть $i = 0.05N = 500$ элемент. Это будет квантиль $z_{0.05}$.

In [63]:
```
#grid = np.linspace(0,10000, 10000)
#ecdf = ECDF(K[:len(K)])  # Empirical Cumulative Distribution Function
#emp_cdf = ecdf(grid)
plt.figure(figsize = (10,10))
#plt.plot(grid,emp_cdf)
plt.hist(K, bins=100, range=(0,1000))
plt.title('ECDF of K distribution')
plt.show()
```



In [64]:
```
print(K[500])
```

```
1.138530742365731
```

Т.е. получили критерий $S = \{K \leq 1.1385\}$. Рассмотрим данную в задаче выборку и посчитаем критерий на ней.

In [65]:
```
K_X = scipy.integrate.quad(f0,0,+np.inf,args=(X,))[0]/scipy.integrate.quad(f
1,0,+np.inf,args=(X,))[0]
```

In [66]:
```
print('K =', K_X)
```

```
K = 140.0601669581188
```

Вывод: так как получившееся значение статистики K >> z_α, то мы принимаем гипотезу H_0. Значит, с помощью байесовского критерия мы смогли проверить гипотезу о распределении выборки и выяснили, что данная выборка имеет нормальное распределение $N(0, \sigma)$.

```
In [ ]:
```