

- JULIAN AUGUSTO CORTES GOMEZ - COD: 1803147
- UMNG - INGENIERIA FORENSE
- PROF. ALEJANDRO CASTELLANOS

```

from google.colab import drive
drive.mount('/content/drive')
from random import random
from scipy import stats
!pip install matplotlib
import matplotlib.pyplot as plt
import os
import pandas as pd
!pip install fitter
!pip install xlrd
from fitter import Fitter, get_distributions, get_common_distributions

```

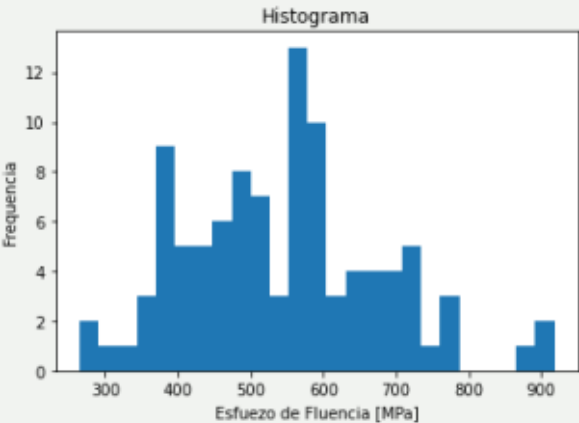
```

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.r
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-packages (3.5.3)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.7/dist-packages (3.0.9)
Requirement already satisfied: cyclizer>=0.10 in /usr/local/lib/python3.7/dist-packages (0.10.0)
Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.7/dist-packages (1.21.0)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist-packages (2.8.2)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (1.4.5)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packages (4.1.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (1.16.0)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/
Requirement already satisfied: fitter in /usr/local/lib/python3.7/dist-packages (1.4.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (1.21.0)
Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-packages (1.3.5)
Requirement already satisfied: easydev in /usr/local/lib/python3.7/dist-packages (0.1.0)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-packages (3.5.3)
Requirement already satisfied: joblib in /usr/local/lib/python3.7/dist-packages (1.1.0)
Requirement already satisfied: scipy>=0.18 in /usr/local/lib/python3.7/dist-packages (1.7.3)
Requirement already satisfied: click in /usr/local/lib/python3.7/dist-packages (8.0.4)
Requirement already satisfied: colorama in /usr/local/lib/python3.7/dist-packages (0.4.4)
Requirement already satisfied: colorlog in /usr/local/lib/python3.7/dist-packages (6.6.2)
Requirement already satisfied: pexpect in /usr/local/lib/python3.7/dist-packages (4.8.0)
Requirement already satisfied: cyclizer>=0.10 in /usr/local/lib/python3.7/dist-packages (0.10.0)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist-packages (2.8.2)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.7/dist-packages (3.0.9)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (1.4.5)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packages (4.1.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (1.16.0)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (2022.1)
Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.7/dist-packages (0.7.0)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/
Requirement already satisfied: xlrd in /usr/local/lib/python3.7/dist-packages (1.1.0)

```

# Caso 1: Propiedades de un material

Usted realiza 100 ensayos de tensión para determinar el esfuerzo de fluencia de una acero 4140. La medida de esta propiedad tiene le comportamiento que se muestra en la figura. Para un diseño de una pieza se establece el esfuerzo de fluencia mínimo en 400 MPa, determine la probabilidad de que el material de la pieza no cumpla con la resistencia deseada.

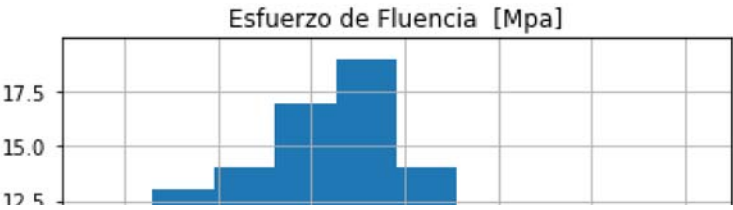


```
archivo = '/content/drive/MyDrive/Datos.xlsx'
df = pd.read_excel(archivo, sheet_name='Caso1')
df.describe()
```

	Esfuerzo de Fluencia [Mpa]	
count	100.000000	
mean	544.642391	
std	135.052183	
min	265.005088	
25%	443.460679	
50%	551.909467	
75%	622.837458	
max	917.492766	

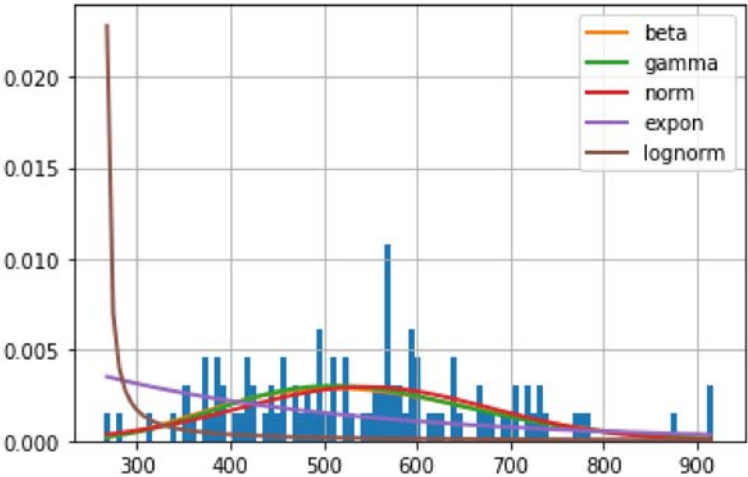
```
df.hist()
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7fafbb37e8d0>]],
      dtype=object)
```



```
f = Fitter(df, distributions=['gamma','lognorm','beta','expon','norm'])
f.fit()
f.summary()
```

	sumsquare_error	aic	bic	kl_div	
beta	0.000250	1368.595460	-1271.469685	inf	
gamma	0.000251	1371.396014	-1275.784370	inf	
norm	0.000251	1376.407981	-1280.119568	inf	
expon	0.000396	1364.032245	-1234.801837	inf	
lognorm	0.001040	1736.097382	-1133.567899	inf	



```
f.get_best(method='sumsquare_error')

{'beta': {'a': 4.903736596010129,
          'b': 11.61718725546137,
          'loc': 178.75133994774788,
          'scale': 1232.666495391395}}

f.fitted_param["beta"]

(4.903736596010129, 11.61718725546137, 178.75133994774788, 1232.666495391395)

a = f.fitted_param["beta"][0]
b = f.fitted_param["beta"][1]
loc = f.fitted_param["beta"][2]
scale = f.fitted_param["beta"][3]
import numpy as np
x = np.linspace(stats.beta(a,b,loc,scale).ppf(0.0001),
```

```

stats.beta(a,b,loc,scale).ppf(0.9999),
1000)

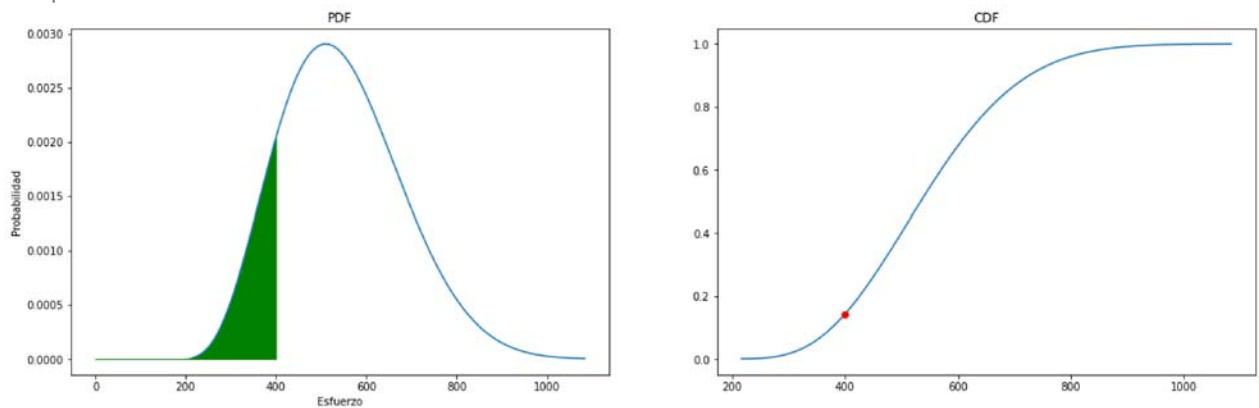
dist = stats.beta(a,b,loc,scale)
valor = 400
fig,(ax1,ax2) = plt.subplots(1,2,figsize=(20,6))
ax1.plot(x, dist.pdf(x))
px = np.arange(0, valor, 0.001)
ax1.fill_between(px, dist.pdf(px), color = 'g')
ax1.set_title("PDF")
ax1.set_xlabel('Esfuerzo')
ax1.set_ylabel('Densidad')

fda_beta = dist.cdf(x)

ax2.plot(x, fda_beta)
ax2.plot(valor,dist.cdf(valor),"ro")
ax2.set_title("CDF")
ax1.set_xlabel('Esfuerzo')
ax1.set_ylabel('Probabilidad')

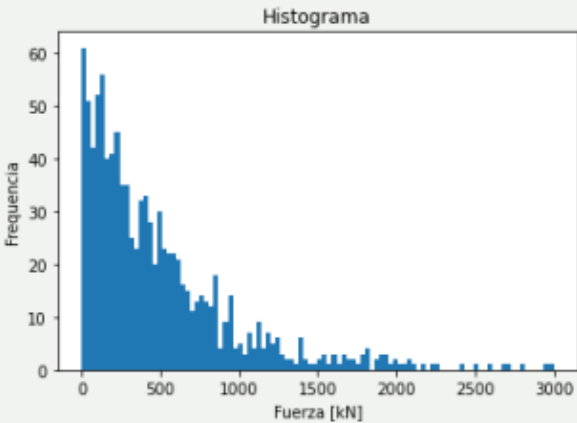
prob1 = dist.cdf(valor)
print(f'La probabilidad de sacar menos de {valor}', f'es: {prob1}')
```

La probabilidad de sacar menos de 400 es: 0.14252938006727864




# Caso 2: Fuerza de una Prensa

Una prensa mecánica presenta fallas de forma constante, por lo que se realiza un estudio con respecto a la fuerza que aplica en cada proceso. Se realizan 1000 pruebas registrando la fuerza que genera la prensa. De acuerdo con el fabricante, la prensa soporta fuerzas hasta de 1300 kN. Con base en los datos, determine la probabilidad de que la prensa exceda la fuerza permitida.



```
archivo = '/content/drive/MyDrive/Datos.xlsx'
df = pd.read_excel(archivo, sheet_name='Caso 2')
df.describe()
```

	Fueza [kN]	
count	1000.000000	
mean	485.928757	
std	481.054123	
min	0.490356	
25%	144.741650	
50%	347.637513	
75%	652.556676	
max	2997.212340	

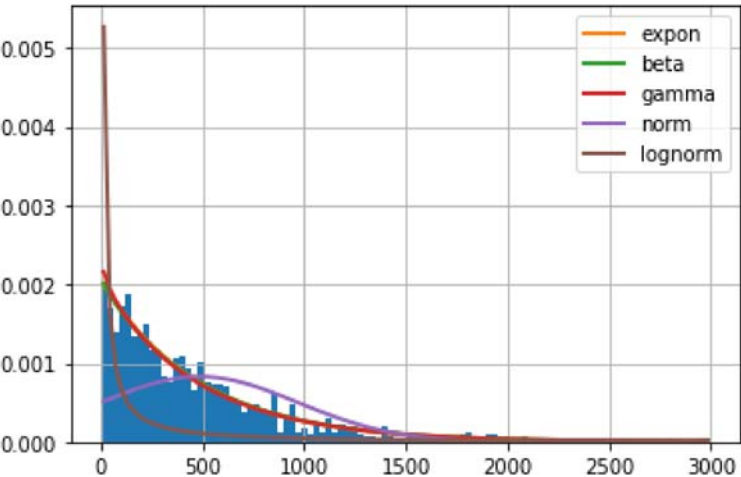
```
df.hist()
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7fafbb274cd0>]],
      dtype=object)
```



```
f = Fitter(df, distributions=['gamma','lognorm','beta','expon','norm'])
f.fit()
f.summary()
```

	sumsquare_error	aic	bic	kl_div	
expon	9.297130e-07	1858.333291	-20782.329664	inf	
beta	9.322977e-07	1856.695012	-20765.737886	inf	
gamma	1.015085e-06	1852.661219	-20687.570441	inf	
norm	1.067366e-05	2190.374970	-18341.670866	inf	
lognorm	2.701613e-05	2031.982822	-17406.108576	inf	



```
f.get_best(method='sumsquare_error')

{'expon': {'loc': 0.4903556588589409, 'scale': 485.43840138541145}}
```

```
f.fitted_param["expon"]

(0.4903556588589409, 485.43840138541145)
```

```
loc = f.fitted_param["expon"][0]
scale = f.fitted_param["expon"][1]
```

```
import numpy as np
x2 = np.linspace(stats.expon(loc, scale).ppf(0.0001),
                  stats.expon(loc, scale).ppf(0.9999),
                  1000)
```

```
dist2 = stats.expon(loc, scale)
```

```

valor = 1300
fig,(ax1,ax2) = plt.subplots(1,2,figsize=(20,6))
ax1.plot(x2, dist2.pdf(x2))
px2 = np.arange(valor,4500, 0.001)
ax1.fill_between(px2, dist2.pdf(px2), color = 'g')
ax1.set_title("PDF")
ax1.set_xlabel('Esfuerzo')
ax1.set_ylabel('Densidad')

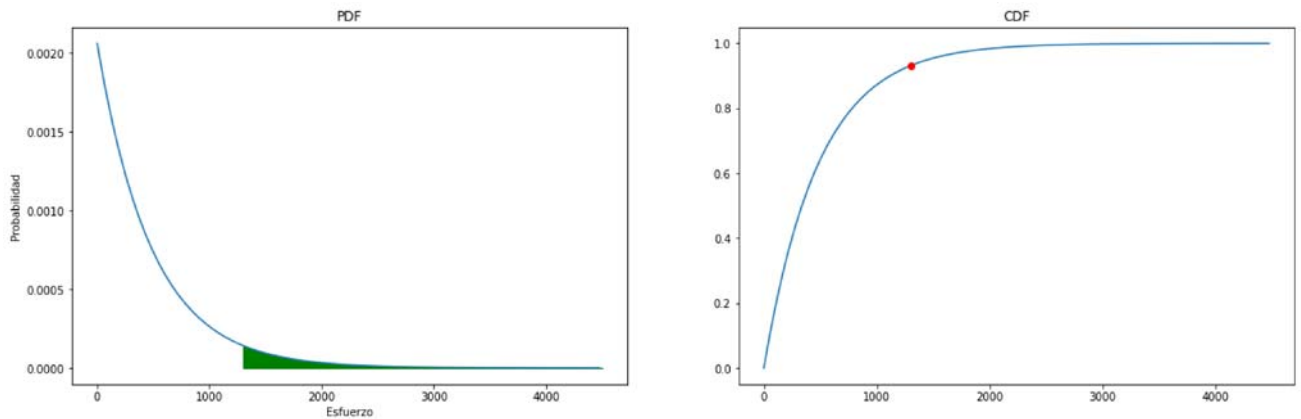
fda_expon = dist2.cdf(x2)

ax2.plot(x2, fda_expon)
ax2.plot(valor,dist2.cdf(valor),"ro")
ax2.set_title("CDF")
ax1.set_xlabel('Esfuerzo')
ax1.set_ylabel('Probabilidad')

prob1 = dist2.cdf(valor)
print(f'La probabilidad de sacar mas de {valor}kN', f'es: {1-prob1}')

```

La probabilidad de sacar mas de 1300kN es: 0.06877042123805688



```

!wget -nc https://raw.githubusercontent.com/brpy/colab-pdf/master/colab_pdf.py
from colab_pdf import colab_pdf
colab_pdf("ForenseMiNIp.ipynb")

```

VlyxbamCaEEEIICSvQKCZ+Z9GixfLnn9vkzp27cuL4Cfn++xmMKSaEEEJImIJGMfE7adOm1Rw5c0iOH  
NklZqyYcvbsGXn8iKtPEEIIISTsQKOY+J1s2d7RE+0KFyksdfUaxbHk0eNHxreEEEEIIa8eGsXE7/z+  
+++yZPES/flh3nx580CBRIoYyfiWEEIIIETVQ6OY+J2nT5/pJdiePXsmCRM1lBYtmku8+PGMbwkhhBB  
CXj00ionfiRkjhsSNG1fixIkjUSJHlsOHD8vqVav1z61/GlcQQgghhLxaaBQTV7N161aZMmWq/P33P7  
J8+QpZvGixHDx4SM6cPWtc4Zrbt2/LuLHjpGGDRjKg/wC5ePGi8Q0hhBBCiG+gUuZ8zsWL12TsuG+kZ  
68e8tXXYyRV6lRsr35dqV69mnGfCxBysXTJUokRI4ZMnTZFevXuJfHiMfSCEEIIIB6FRjHx09i97pEf  
f5Ftf26TdWvYxc0bNyVChAjGt665f/++nDp1Sv+7d0ky2b9/f4jf3r1zR3799Vf5Yd4PcuHCBems8ZZ  
pW/6TiRuPuf38fvSy8QtCCCHKzYBGMfE7derW1h5fhFEcP35cqlWvpmOMrXD//g05fPmyXL16VSJHii  
QrV6yUQ4cOGd8qAQ4fXmLHjiPxEySQiJEiGmeJt4z85bAMWxNA7WfNfoawEEIIebOgUUz8TooUKSRz5  
sySIUMGqVmzpqR+07U2Zq0QNWoU7WkuULCALC5TWpIkSarjku2iRYsm7733Py1VqqQkiJ/AOEsIYYQQ  
4hk0ionfmT7te1m4YKH8/PPPEiNmDBk/7lvLk+Vg9GbPkUNOnzote/7ZI+fPn50UysgmhBBCPElNIq  
J3zmwf7+0btNaIkeOIlgIRNGhFFi32ColS5aQcOHCyAZNmyV37tzy7v/eNb4hhBBCPENNIqJ38mQKa  
Os37BBL164IASXLtKhE9GiRjO+dQ/WN67foL507tJJypQtIxEjMnaYEEIIIB6FRjHx06VK1ZL48eNLw  
oQJtWFct14diRvP2kQ7QgghhJCXAY1i4nemTZ0mhQp9IF+O+FLatW8n2bJ1szzRjhBCCCHKZUDLhPg  
rB4x4/uZemvNnb+skZ9X/yx37941viWEEIIIEfXQKCZ+49GjR/LgwQPJnDmTPHnyRI4dOxb8efz4sXE  
VIYQQQsirr0Yx8Rv//LNHft3wq5QtV1bix0jLVq2kKbNmUoPvMeEEEEIIWEGFsXeb9y4cUMuXbokkS  
JFkl07d0nUqFGDP1hiJRBCCEKREcjmPiNx48eyd69+3QM8ZmZ/W/5ocxxYQQQggJS4S7cev6M+NvQ  
nzKwQMHZfnyFXqlCcQQ264v3LxFM73+sC9p1aK1zJ3szzh6vblx75HkHPiLceScTEliStlsyYyj0DP+  
1yPy6MnLVQkRwoeTo0PKG0fO+Wxfewk1c6dxFHq29iwhSeNENY7CFiev3JFFu84YR84pnDGH5Eod3zg  
i3lK7dm2ZNGWicUQICVRoFJM3hka0it8EaBS/yJYj16X+5D+NI+f0rpBFWhZKaxwRb6FRTAgBDJ8ghB  
BCCCEBD41iQgghhBAS8NAoJoQQQgghAQ+NYKIIIIYQQEvDQKCaEEEEIIIEPjWJCCCGEELw0CgmhBBC  
CEBD41iQgghhBAS8NAoJoQQQgghAQ+NYvJacOPGDfmwZi0ZOWKUcYYQQgghXhFQKCZhnkePHsmC+Qv1  
nXeyypMnT4yzhBBCCG+I9yNW9efGX8TEib5c+ufsmvXLkmRIoUcOHBQunXvanwjcVpMTfnt19/k5Mm  
TsnLFKtm0aZPxzevNjXuPJ0fAX4yJN5sI4cPJ0SHljSPn/LLvvLSaudM4Cj1be5aQpHGikf02X3qmv  
z09znj60Vw9vo9WbX3vHHknN4VskjLQmmNI+IttWvXlK1TJhpHhJBahUYxCdNcvXJv5s+fL/ny55NLF  
y/JX3/tDmEUP3z4UE6dPCU3LHE8/IvshmlFCuOb1xsaxS/yqozi+TtOSbeF/xhHYQsaxb6BRjEhBDB8  
goRprly5Inv27JVBAwbJkMFDZPWqVfLbb78Z34pEjhxZ0qVPJ++99z+JFSuWcZYQQgghXDNofJmWtYa  
MGWtC+LgYe0li6dWnt5QtV06KFCliFeSIIYQQ4htoFJPXhntP0skHHxQ0jgghhBBCfAeNYvLakDZtGi  
lQsIBxRAgghBDi02gUE0IIIIYSQgIdGMSGEEEEIICXhoFBNCCCGEKICHRjEhhBBCCAl4aBQTQgghhJCAh  
0YxIYQQQggJgGUE0IIIIYSQgCfcjVvXnx1/E/Ja06pFa5k3b55xHb5dsMROXH1rnHkmIdPnsriXWeM  
ozebcOpT00/KoAMXnL52TzYfuWwchZ4qOZNLtMgRjCPnHLt8R7b9d9U4Clv0rpBFWhZKaxw5Z+Qvh+T  
SrQfGkX0yvRVHGuRlBwFDrVr15ZJUyYaR4SQQIVGMXlJef2M4hrjf5edJ68ZR4R4j1WjuPJIX+XYpT  
vGkXPkZU5q3zbIZRwFDjSKCSGA4ROEEEEIIISTgoVFMCCGEEEEICHhrFhBBCCCEk4KFRtAgghhBBCAh4ax  
YQQQgghJOChUuWIIYQQQgIEgSWEEEEIIISTgoVFMCCGEEEEICHhrFJMxz7949uXbtuly/fl0ePnxonCWE  
EEII8R00ikmYBgxb0iVLpX+/tK3Tz+ZO2euPHr0yPiWEEIIICQ30CgmYZpIkSJJ6dK1ZcTI4dKjZ3f  
ZuX0XXLP0yfiWEEIIICQ3hLtx6/oz429CwjSHDX2WswPHyaDPBkrCuHH10YRVrFyxQo4ePsqbNm6WP/  
/8U5+3wne/HZV9Z24YR875qu7/JHy4cMZR6Kkx/nfZefKacUSI9/SukEvaFkprHDmn+Mhf5di108aRc  
8p1SyrFNShlHIUtVv5zTlbtPWccuearOqrOhrdeZ2vXri2Tpkw0jgghgQo9xeS14M6d09ogr1K1isSJ  
E8c4KxIrVkwPxaa0NGnaRJILS2actca2/67KT6qhdFchhLx6D1245bB+OvrQ00MI8QYaxSTMc/nSZRk  
29AspXryYlChRXMLZeG0jRowoiRIlklSpUknUqFGNs4QQQgghnkGjmIRp7t+/LzNmzJTDhw/LgwcPZP  
36DXLjhvuQB0IIIIYQQT6BRTMI8efLm1lq1a0n06NElYsQIITzFhBBCCCG+gEYxCdMgJKJQoUJSo0Z1q  
VS5khQuXFhix45tfEsIIYQQ4htoFBNCCCGEKICHRjEhhBBCCAl4aBQTQgghhJCAh0YxIYQQQggJgGUE  
E0IIIIYSQgIdGMSGEEEEIICXhoFBNCCCGEKIAN3I1b171NPHkjaNWitcybN884ck+z6dtl/cGLxpFzimV  
K5NMNq3YevyY37j8yJgjxnvsJY0qq+NGNI+dsPXZF7j58Yhw5p1y2pPJtg1zG0cvh2KXb8vmKA8aRc3  
Dd8St3jSPXHB1SXiKEt15na9euLZOmTDSOCCGBCo1i8sbgL60YkEDhVRjFf5+6L1XGbTGOFAONYKKIN  
zB8ghBCCCGEBDw0igkhhBBCSMBDo5gQQgghhAQ8NIoJIIYQQQkjaQ60YEEIIISQAUxvnmVy//kyePeOa  
C4BGMSGEEEEIAHHP4jPp3fWBVCh5X6qWvS8tGz2QHdvcL9v4pk0jmBBCCCEkQLh37510aPtAtmx6Kqa  
D+Mi/z6RHp4eyb09gG8Y0igkhhBBCAoQ1q5/I2TMvhks8fiwy63v1vwCGRjEJ8zx48EBOnDiherJH5M  
aNG8ZZQgghhHjK0X+fGn+9yJHDgr1bTKOYhHm2bPldvvt2gkyd0k3mzpmrjWRCCCGEE7iJM53e0yaz  
PpOkG8iNIpJmObOnTuyc8d0qVChvHTu3EnOnTsvx44dM741hBBCXk8uymPZILdlSvYX1XJTjstD4xv/  
Uq5iRIkW3Tiwo0btimZfgUm4G7eucx00EmY5d+6czJo5WypVqiipUqeS76d/LxkzZZISJYrr769euSo  
//PCDHDhwUDZv2ixFiHbR561w6NxnX7vkXfkjTt370qM6E60ySsE6Yqu0hXw+vhhMV33HzYUiBEiSM  
SIEYwzYYOHDx9K+PDhVbpeXaMUP0YkyZAktnEUxIP7D3S6IkW0ZJzxLbfvP5Z9Zz0Li3r8+Ik8fvJEO  
kaJbJwJSd40CSScB0K3c8c02bh5o3FEiP85K4+0QfzMTk7fexZNskpU48h/7N/7VAb1fSjnzWwZgNGi  
ibRoE4lGMY1iEpY5e/aszJ41J9gonvH9DEmfIYOUFLCf4+1FWFMPH78WP8dzpOW0Au6de0ugwYNlKj  
R/K+0PKFnj17Sp29viREjhnEmbNC3T1/p0rWLxIkTxzjz6pk5c5ZkUh2rvHnzGGfCBnNmz5U0adNI/v



