Llamar al archivo de fuente de datos

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

datos = pd.read_excel('C:/Users/Villamar/Desktop/JOSS/SEXTOSEMESTRE/MODELAMI
datos


```

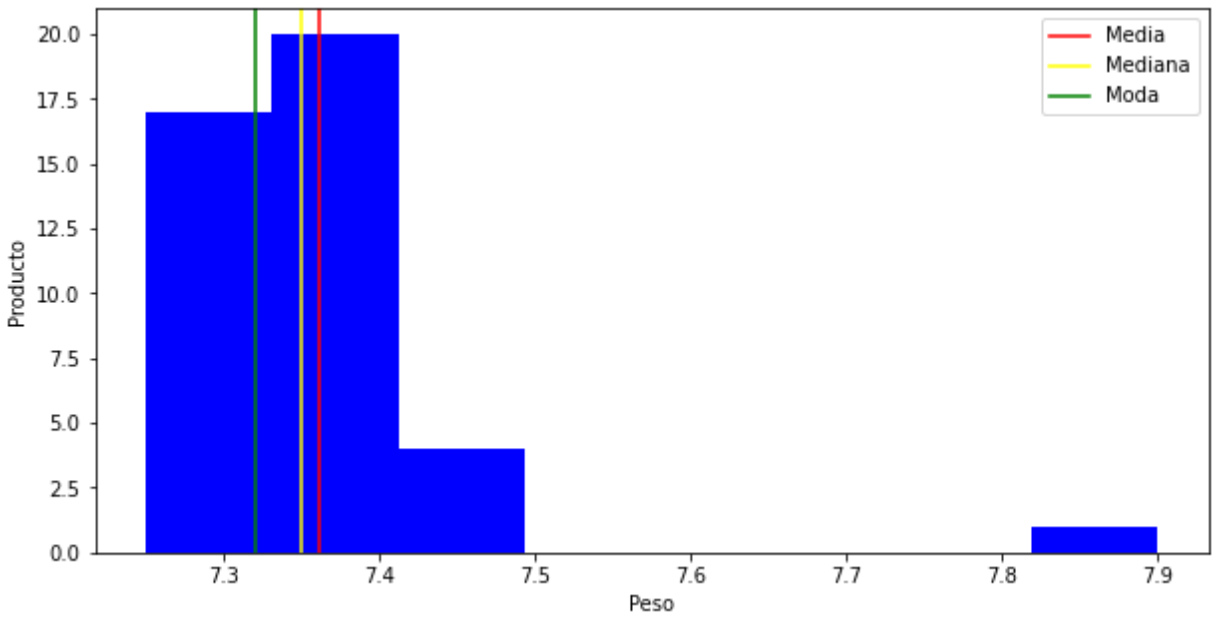
Out[1]:

	articulos	peso_libra
0	1	7.90
1	2	7.31
2	3	7.36
3	4	7.31
4	5	7.35
5	6	7.32
6	7	7.48
7	8	7.37
8	9	7.32
9	10	7.28
10	11	7.36
11	12	7.32
12	13	7.37
13	14	7.33
14	15	7.36
15	16	7.35
16	17	7.34
17	18	7.33
18	19	7.26
19	20	7.33
20	21	7.38
21	22	7.30
22	23	7.25
23	24	7.35
24	25	7.34
25	26	7.40
26	27	7.40
27	28	7.27
28	29	7.32
29	30	7.42

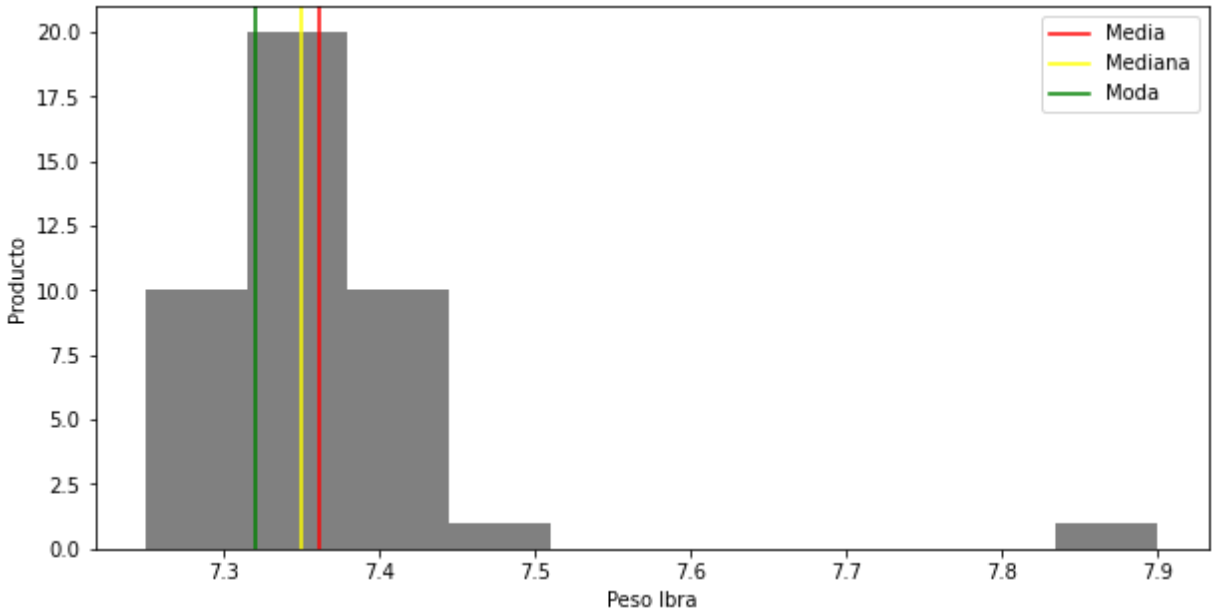
	articulos	peso_libra
30	31	7.39
31	32	7.41
32	33	7.44
33	34	7.30
34	35	7.36
35	36	7.37
36	37	7.43
37	38	7.28
38	39	7.39
39	40	7.35
40	41	7.29
41	42	7.41

Histograma

In [3]:
```
x=datos["peso_libra"]
plt.figure(figsize=(10,5))
plt.hist(x,bins=8,color='blue')
plt.axvline(x.mean(),color='red',label='Media')
plt.axvline(x.median(),color='yellow',label='Mediana')
plt.axvline(x.mode()[0],color='green',label='Moda')
plt.xlabel('Peso')
plt.ylabel('Producto')
plt.legend()
plt.show()
```



In [4]:
```
x=datos["peso_libra"]
plt.figure(figsize=(10,5))
plt.hist(x,bins=None,color='grey')
plt.axvline(x.mean(),color='red',label='Media')
plt.axvline(x.median(),color='yellow',label='Mediana')
plt.axvline(x.mode()[0],color='green',label='Moda')
plt.xlabel('Peso lbra')
plt.ylabel('Producto')
plt.legend()
plt.show()
```



Mediana

In [7]:
```
# enviando las medias a t1, t2, t3 para su utilización
print("Mediana:", )
t1 = datos.median()
print( "la Mediana de pesos es: ", t1)
print("DIRECTAMENTE DEL DATAFRAME ")
datos.median()
```

```
Mediana:
la Mediana de pesos es:  articulos     21.50
peso_libra     7.35
dtype: float64
DIRECTAMENTE DEL DATAFRAME
```

Out[7]:
```
articulos     21.50
peso_libra     7.35
dtype: float64
```

Media

In [8]:
```
# enviando las medias a t1, t2, t3 para su utilización
print("Media:", )
t1 = datos.mean()
print( "la Media de pesos: ", t1)
print("DIRECTAMENTE DEL DATAFRAME ")
datos.mean()
```

```
Media:
la Media de pesos:  articulos     21.500000
peso_libra     7.361905
dtype: float64
DIRECTAMENTE DEL DATAFRAME
```

Out[8]:
```
articulos     21.500000
peso_libra     7.361905
dtype: float64
```

Moda

In [11]:
```
# enviando las modas a mo1, mo2, mo3 para su utilización
print("Moda:")
mo1 = datos["peso_libra"].mode()
print( "la Moda de peso de articulos: ", mo1)
pd.DataFrame(mo1)

```

```
Moda:
la Moda de peso de articulos:  0    7.32
1    7.35
2    7.36
dtype: float64
```

Out[11]:

	0
0	7.32
1	7.35
2	7.36

In [12]:
```
# Tomamos los datos de las columnas
datos[['peso_libra']].describe()
# describe(), nos presenta directamente la media, desviación standar, el val
#o, valor máximo, el 1er cuartil, 2do Cuartil, 3er Cuartil
```

Out[12]:

	peso_libra
count	42.000000
mean	7.361905
std	0.099001
min	7.250000
25%	7.320000
50%	7.350000
75%	7.387500
max	7.900000

In [13]:
```
# seleccionamos los datos del mes de abril 30 registros para calcular estadi
df_1 = datos[:30]
print ("Estadisticos de 30 REGISTROS")
print ("-----------------------------")
df_1[['peso_libra']].describe()
```

```
Estadisticos de 30 REGISTROS
-----------------------------
```

Out[13]:

	peso_libra
count	30.000000
mean	7.359333
std	0.112769
min	7.250000
25%	7.320000
50%	7.340000
75%	7.367500
max	7.900000

In [14]:
```
df_2 = datos[10:21]
df_2
```

Out[14]:

	articulos	peso_libra
10	11	7.36
11	12	7.32
12	13	7.37
13	14	7.33
14	15	7.36
15	16	7.35
16	17	7.34
17	18	7.33
18	19	7.26
19	20	7.33
20	21	7.38

In [15]:
```
print ("Estadisticos del mes de Mayo ")
print ("-----------------------------")
df_2[['peso_libra']].describe()
```

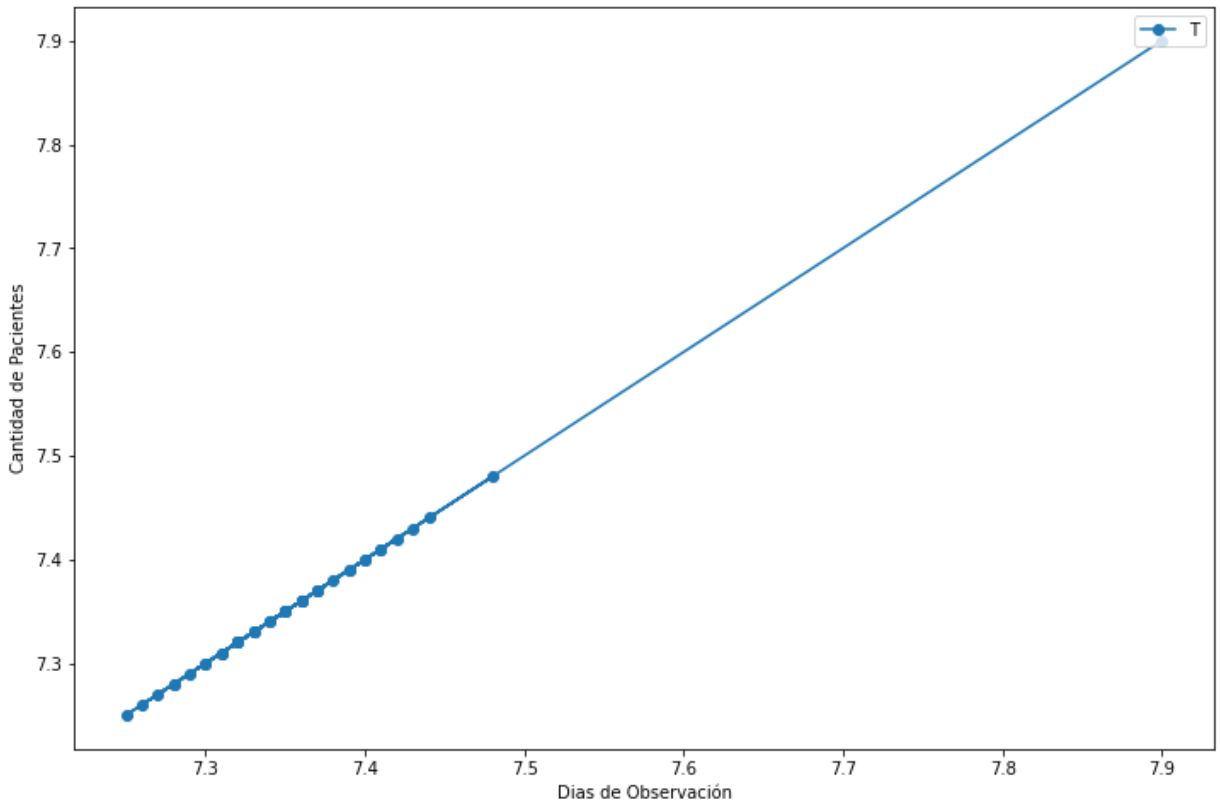
```
Estadisticos del mes de Mayo
-----------------------------
```

Out[15]:

	peso_libra
count	11.000000
mean	7.339091
std	0.032390
min	7.260000
25%	7.330000
50%	7.340000
75%	7.360000
max	7.380000

In [17]:
```
#x = np.arange(61)
x = datos["peso_libra"]
t1 = datos["peso_libra"]
plt.figure(figsize=(12,8))
plt.plot(x,t1,marker='o')
plt.xlabel('Dias de Observación')
plt.ylabel('Cantidad de pesos')
plt.legend(('TOTAL PESOS'), prop = {'size':10},loc='upper right')
```

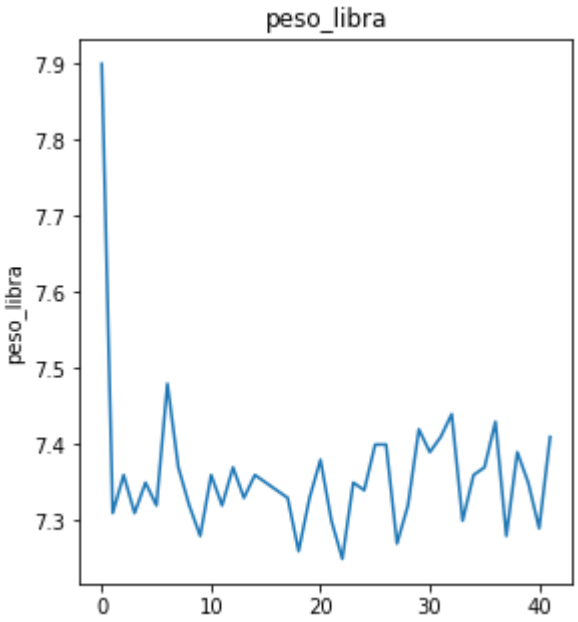
Out[17]: <matplotlib.legend.Legend at 0x23724e67d30>



In [22]:
```
x = range(42)
plt.figure(figsize=(15,5))
plt.subplot(131)
t1 = datos["peso_libra"]
p1, = plt.plot(x,t1)
plt.ylabel('peso_libra')
plt.title('peso_libra')

```

Out[22]: Text(0.5, 1.0, 'peso_libra')



In [23]:
```
# OBTENER LOS DATOS UNICOS DE LA TABLA
lis = datos["peso_libra"].unique()
lis
dfclases=pd.DataFrame(lis,columns=["peso_libra"])
dfclases
```

Out[23]:

	peso_libra
0	7.90
1	7.31
2	7.36
3	7.35
4	7.32
5	7.48
6	7.37
7	7.28
8	7.33
9	7.34
10	7.26
11	7.38
12	7.30
13	7.25
14	7.40
15	7.27
16	7.42
17	7.39
18	7.41
19	7.44
20	7.43
21	7.29

Frecuencias Absolutas

In [24]:
```
#TABLA DE FRECUENCIAS ABSOLUTAS
# OBTENER FRECUENCIAS ABSOLUTAS DE CADA CLASE
datafi=pd.crosstab(index=datos["peso_libra"], columns = "Fi")
# Creamos una lista con los valores de las frecuencias
li = datafi.values
# agregamos una columna al dataframe
dfclases["Fi"] = li
#observamos dfclase
dfclases
```

Out[24]:

	peso_libra	Fi
0	7.90	1
1	7.31	1
2	7.36	1
3	7.35	2
4	7.32	1
5	7.48	2
6	7.37	2
7	7.28	4
8	7.33	3
9	7.34	2
10	7.26	4
11	7.38	4
12	7.30	3
13	7.25	1
14	7.40	2
15	7.27	2
16	7.42	2
17	7.39	1
18	7.41	1
19	7.44	1
20	7.43	1
21	7.29	1

Frecuencias Relativas

In [25]:
```
# Columna de Frecuencia relativa
total = dfclases.sum(axis=0)
datahi = dfclases["Fi"]/total["Fi"] # aqui calculamos la frecuencia
datahi.values
# agregamos nueva columna de frecuencia relativa
dfclases["hi"] = datahi
dfclases
```

Out[25]:

	peso_libra	Fi	hi
0	7.90	1	0.023810
1	7.31	1	0.023810
2	7.36	1	0.023810
3	7.35	2	0.047619
4	7.32	1	0.023810
5	7.48	2	0.047619
6	7.37	2	0.047619
7	7.28	4	0.095238
8	7.33	3	0.071429
9	7.34	2	0.047619
10	7.26	4	0.095238
11	7.38	4	0.095238
12	7.30	3	0.071429
13	7.25	1	0.023810
14	7.40	2	0.047619
15	7.27	2	0.047619
16	7.42	2	0.047619
17	7.39	1	0.023810
18	7.41	1	0.023810
19	7.44	1	0.023810
20	7.43	1	0.023810
21	7.29	1	0.023810

In [26]:
```
total1 = dfclases.sum(axis=0) # totales
total1
```

Out[26]:
```
peso_libra    162.28
Fi             42.00
hi              1.00
dtype: float64
```

In [27]:
```
# la suma de las frecuencias Relativas nos da 1
# aqui vamos a calcular la frecuencia absoluta
FA = dfclases["Fi"].values
# obtenemos FA
a=[]
b=0
for c in FA:
 b = c + b
 a.append(b)
dfclases["FA"] = a
HI = dfclases["hi"].values
# obtenemos HI
a=[]
b=0
for c in HI:
 b = c + b
 a.append(b)
dfclases["HI"] = a
dfclases
```

Out[27]:

	peso_libra	Fi	hi	FA	HI
0	7.90	1	0.023810	1	0.023810
1	7.31	1	0.023810	2	0.047619
2	7.36	1	0.023810	3	0.071429
3	7.35	2	0.047619	5	0.119048
4	7.32	1	0.023810	6	0.142857
5	7.48	2	0.047619	8	0.190476
6	7.37	2	0.047619	10	0.238095
7	7.28	4	0.095238	14	0.333333
8	7.33	3	0.071429	17	0.404762
9	7.34	2	0.047619	19	0.452381
10	7.26	4	0.095238	23	0.547619
11	7.38	4	0.095238	27	0.642857
12	7.30	3	0.071429	30	0.714286
13	7.25	1	0.023810	31	0.738095
14	7.40	2	0.047619	33	0.785714
15	7.27	2	0.047619	35	0.833333
16	7.42	2	0.047619	37	0.880952
17	7.39	1	0.023810	38	0.904762
18	7.41	1	0.023810	39	0.928571
19	7.44	1	0.023810	40	0.952381
20	7.43	1	0.023810	41	0.976190
21	7.29	1	0.023810	42	1.000000

Genere 10 números aleatorios por el método congruencial multiplicativo

determine lo valores en peso en libras de cada uno.

Xn+1 = (1140671485Xn+C) mod(M) Xn=81, C=12820163; M= 264

In [33]:
```
# - Xn+1 = (1140671485Xn+C) mod(M) Xn=81, C=12820163; M= 264
n, m, a, x0, c = 10, 264, 1140671485, 81, 12820163
x = [1] * n
r = [0.1] * n
print (" Generador Congruencial multiplicativo")
print ("-------------------------------------")
for i in range(0, n):
    x[i] = ((a*x0)+c) % m
    x0 = x[i]
    r[i] = x0 / m
d = {'Xn': x, 'ri': r }
df1 = pd.DataFrame(data=d)
# df1.head()
df1
```

```
 Generador Congruencial multiplicativo
-------------------------------------
```

Out[33]:

	Xn	ri
0	80	0.303030
1	259	0.981061
2	162	0.613636
3	101	0.382576
4	196	0.742424
5	87	0.329545
6	62	0.234848
7	49	0.185606
8	0	0.000000
9	59	0.223485

In [ ]:
```
1
```