Fuentes de datos

Los datos pueden tener diversas fuentes a las que se puede acceder directa o indirectamente. La calidad, pertinencia y veracidad de un dato depende en gran medida de su origen y el modo en el que éste se obtiene. Una de las bases del método científico es la de poder identificar, aislar y de ser posible, reproducir la fuente de los datos que son objetos de un estudio

Datos continuos y Datos discretos.

Los datos cuantitativos se dividen en datos continuos que puede admitir cualquier valor intermedio dentro de un intervalo de los números reales. Por ejemplo el rendimiento académico, la estatura, el peso, salario, tiempo, volumen que pueden ser obtenidos por medición. Los datos discretos son aquella que admite interrupciones verdaderas en su medición y por lo tanto no admite valores intermedios. Por ejemplo, número de alumnos de la clase, el número de hijos de una familia, el número de clientes en espera en un almacén.

Observación estadística de los datos

Por ahora tenemos en forma natural los datos obtenidos del registro de pedidos e incluso la region constituyen una fuente de mucha información, los registros en forma de tabla tienen los siguientes metadatos:

- Fecha de orden
- Region
- Rep
- Item
- Units
- Unit Cost
- Total

La Distribución de frecuencias separa los datos en clases y muestra el número de ocurrencias en cada clase, o frecuencia de clase, este tipo de agrupación facilita la interpretación y la presentación en cuadros numéricos, que luego se representan en gráficos.

Histogramas

En Python, podemos graficar fácilmente un histograma con la ayuda de la función hist de matplotlib, simplemente debemos pasarle los datos y la cantidad de contenedores en los que queremos dividirlos. Aqui vamos a construir un Histograma de frecuencias:

- importamos pandas
- matplotlib
- numpy
- Creamos un DataFrame con Pandas
- Presentamos los datos

1. La biblioteca Matplotlib.

Matlplotib es la biblioteca de graficación basada en Python más popular y sobre la que una gran catidad de proyectos se basan para despliegue de gráficos y visualización de datos.

Ejercicio

Lo primero que se realiza es importar la libreria de pandas, matplotlib y numpy los cuales seran de mucha utilidad para hacer graficos detallados

Luego lo que se realiza es utilizar una variable para poder incluir el archivo HTML que es con el cual se ha estado trabajando, despues, se le añade un header para poder utilizar el encabezado de la tabla en ves de enumerarlas

In [18]:
```
# importamos la libreria Pandas, matplotlib y numpy que van a ser de mucha utilid
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# leemos los datos de la tabla del directorio Data de trabajo
url = 'file:///C:/Users/Villamar/Desktop/JOSS/SEXTOSEMESTRE/MODELAMIENTOSIMULACIO
dfs = pd.read_html(url, header=0)
df = dfs[0]
#Presentamos los datos en un DataFrame de Pandas
df
```

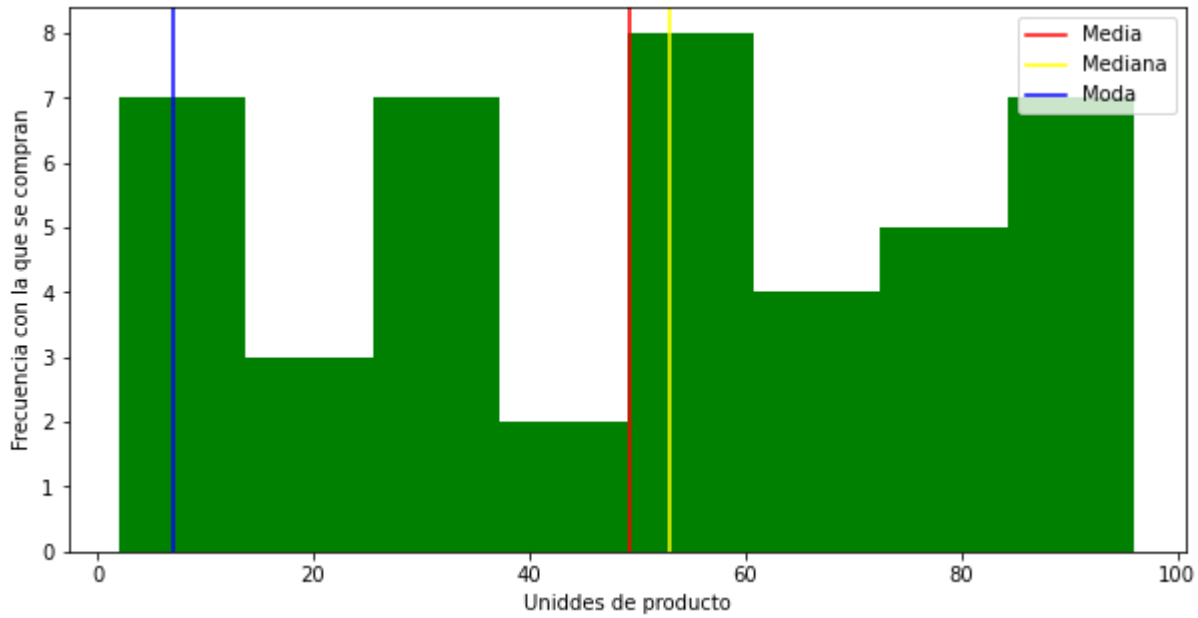
Out[18]:

	OrderDate	Region	Rep	Item	Units	UnitCost	Total
0	1/6/2020	East	Jones	Pencil	95	1.99	189.05
1	1/23/2020	Central	Kivell	Binder	50	19.99	999.50
2	2/9/2020	Central	Jardine	Pencil	36	4.99	179.64
3	2/26/2020	Central	Gill	Pen	27	19.99	539.73
4	3/15/2020	West	Sorvino	Pencil	56	2.99	167.44
5	4/1/2020	East	Jones	Binder	60	4.99	299.40
6	4/18/2020	Central	Andrews	Pencil	75	1.99	149.25
7	5/5/2020	Central	Jardine	Pencil	90	4.99	449.10
8	5/22/2020	West	Thompson	Pencil	32	1.99	63.68
9	6/8/2020	East	Jones	Binder	60	8.99	539.40
10	6/25/2020	Central	Morgan	Pencil	90	4.99	449.10
11	7/12/2020	East	Howard	Binder	29	1.99	57.71
12	7/29/2020	East	Parent	Binder	81	19.99	1619.19
13	8/15/2020	East	Jones	Pencil	35	4.99	174.65
14	9/1/2020	Central	Smith	Desk	2	125.00	250.00
15	9/18/2020	East	Jones	Pen Set	16	15.99	255.84
16	10/5/2020	Central	Morgan	Binder	28	8.99	251.72
17	10/22/2020	East	Jones	Pen	64	8.99	575.36
18	11/8/2020	East	Parent	Pen	15	19.99	299.85
19	11/25/2020	Central	Kivell	Pen Set	96	4.99	479.04
20	12/12/2020	Central	Smith	Pencil	67	1.29	86.43
21	12/29/2020	East	Parent	Pen Set	74	15.99	1183.26
22	1/15/2021	Central	Gill	Binder	46	8.99	413.54
23	2/1/2021	Central	Smith	Binder	87	15.00	1305.00
24	2/18/2021	East	Jones	Binder	4	4.99	19.96
25	3/7/2021	West	Sorvino	Binder	7	19.99	139.93
26	3/24/2021	Central	Jardine	Pen Set	50	4.99	249.50

	OrderDate	Region	Rep	Item	Units	UnitCost	Total
27	4/10/2021	Central	Andrews	Pencil	66	1.99	131.34
28	4/27/2021	East	Howard	Pen	96	4.99	479.04
29	5/14/2021	Central	Gill	Pencil	53	1.29	68.37
30	5/31/2021	Central	Gill	Binder	80	8.99	719.20
31	6/17/2021	Central	Kivell	Desk	5	125.00	625.00
32	7/4/2021	East	Jones	Pen Set	62	4.99	309.38
33	7/21/2021	Central	Morgan	Pen Set	55	12.49	686.95
34	8/7/2021	Central	Kivell	Pen Set	42	23.95	1005.90
35	8/24/2021	West	Sorvino	Desk	3	275.00	825.00
36	9/10/2021	Central	Gill	Pencil	7	1.29	9.03
37	9/27/2021	West	Sorvino	Pen	76	1.99	151.24
38	10/14/2021	West	Thompson	Binder	57	19.99	1139.43
39	10/31/2021	Central	Andrews	Pencil	14	1.29	18.06
40	11/17/2021	Central	Jardine	Binder	11	4.99	54.89
41	12/4/2021	Central	Jardine	Binder	94	19.99	1879.06
42	12/21/2021	Central	Andrews	Binder	28	4.99	139.72

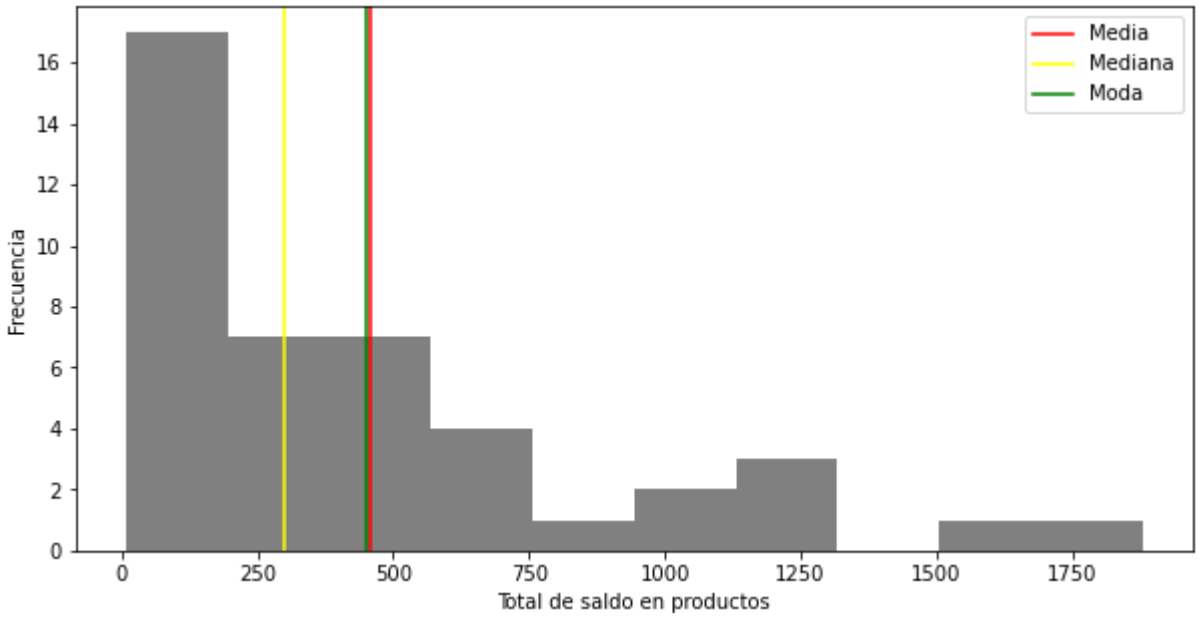
Luego se va a realizar un grafico de las unidades para eso se realiza la siguiente codificacion utilizando los colores, la media, la moda y la mediana

In [65]:
```
x=df["Units"]
plt.figure(figsize=(10,5))
plt.hist(x,bins=8,color='green')
plt.axvline(x.mean(),color='red',label='Media')
plt.axvline(x.median(),color='yellow',label='Mediana')
plt.axvline(x.mode()[0],color='blue',label='Moda')
plt.xlabel('Uniddes de producto')
plt.ylabel('Frecuencia con la que se compran')
plt.legend()
plt.show()
```



El siguiente grafico va a mostrar el total de los datos parecido al grafico anterior pero para este se utilizo otra de las columnas

In [30]:
```
x=df["Total"]
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('Total de saldo en productos')
plt.ylabel('Frecuencia')
plt.legend()
plt.show()
```



En esta parte se envia la mediana de cada uno de los datos de las figuras anteriores para eso se especfica tres variables de las tres columnas de las cuales se relizaron los graficos

In [31]:
```
# enviando las medias a t1, t2, t3 para su utilización
print("Mediana: " )
t1 = df.median()
t2 = df.median()
t3 = df.median()
print( "la Mediana de unidades: ", t1)
print( "la Mediana en total: ", t2)
print( "la Mediana entre el total de unidades: ", t3)
print("DIRECTAMENTE DEL DATAFRAME ")
df.median()
```

```
Mediana:
la Mediana de unidades:  Units        53.00
UnitCost      4.99
Total       299.40
dtype: float64
la Mediana en total:  Units        53.00
UnitCost      4.99
Total       299.40
dtype: float64
la Mediana entre el total de unidades:  Units        53.00
UnitCost      4.99
Total       299.40
dtype: float64
DIRECTAMENTE DEL DATAFRAME
```

Out[31]:
```
Units        53.00
UnitCost      4.99
Total       299.40
dtype: float64
```

En esta parte se envia la media de cada uno de los datos de las figuras anteriores para eso se especifica tres variables de las tres columnas de las cuales se relizaron los graficos

In [33]:
```
# enviando las medias a t1, t2, t3 para su utilización
print("Media:", )
t1 = df.mean()
t2 = df.mean()
t3 = df.mean()
print( "la Media de unidades: ", t1)
print( "la Media en total: ", t2)
print( "la Media entre el total de las unidades: ", t3)
print("DIRECTAMENTE DEL DATAFRAME ")
df.mean()
```

```
Media:
la Media de unidades:  Units        49.325581
UnitCost     20.308605
Total       456.462326
dtype: float64
la Media en total:  Units        49.325581
UnitCost     20.308605
Total       456.462326
dtype: float64
la Media entre el total de las unidades:  Units        49.325581
UnitCost     20.308605
Total       456.462326
dtype: float64
DIRECTAMENTE DEL DATAFRAME
```

Out[33]:
```
Units        49.325581
UnitCost     20.308605
Total       456.462326
dtype: float64
```

Despues se realiza la moda de cada uno de los datos de las figuras anteriores para eso se especfica tres variables de las tres columnas de las cuales se relizaron los graficos lueg mostramos ambas modas con su respectivo reusultado

In [56]:
```
# enviando las modas a mo1, mo2, mo3 para su utilización
print("Moda:")
mo1 = df["Total"].mode()
mo2 = df["Units"].mode()
mo3 = df["UnitCost"].mode()
print( "la Moda del total de unidades: ", mo1)
print( "la Moda de las unidades: ", mo2)
print( "la Moda del osto unitario: ", mo3)
pd.DataFrame(mo1)
pd.DataFrame(mo2)
pd.DataFrame(mo3)
```

```
Moda:
la Moda del total de unidades:  0    449.10
1    479.04
dtype: float64
la Moda de las unidades:  0     7
1    28
2    50
3    60
4    90
5    96
dtype: int64
la Moda del osto unitario:  0    4.99
dtype: float64
```

Out[56]:

	0
0	4.99

A continuacion, se tomaran los dataos de las columnas, se ha usado Units, Total y UniCost por que son datos numericos esto lo usamos seguido de un describe, lo cual ns decribe directamente la media, desviacion estandae, valor minimo, maximo, cada uno de los cuartiles

In [57]:
```
# Tomamos los datos de las columnas
df[['Units','Total','UnitCost']].describe()
# describe(), nos presenta directamente la media, desviación standar, el valor mi
#o, valor máximo, el 1er cuartil, 2do Cuartil, 3er Cuartil
```

Out[57]:

	Units	Total	UnitCost
count	43.000000	43.000000	43.000000
mean	49.325581	456.462326	20.308605
std	30.078248	447.022104	47.345118
min	2.000000	9.030000	1.290000
25%	27.500000	144.590000	3.990000
50%	53.000000	299.400000	4.990000
75%	74.500000	600.180000	17.990000
max	96.000000	1879.060000	275.000000

Seguido vamos a seleccionar los datos estadisticos los primeros 30 registros

In [38]:
```
df_1 = df[:30]
print ("Estadisticos de 30 REGISTROS")
print ("-----------------------------")
df_1[['Units','Total','UnitCost']].describe()
```

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

Out[38]:

	Units	Total	UnitCost
count	30.000000	30.000000	30.000000
mean	52.900000	402.167333	12.277333
std	29.117301	392.037385	22.296454
min	2.000000	19.960000	1.290000
25%	29.750000	153.797500	3.490000
50%	54.500000	253.780000	4.990000
75%	74.750000	479.040000	15.742500
max	96.000000	1619.190000	125.000000

Con estas dos lineas de codigo vamos a mostrar un intervalo de 15 numeros los cuales comenzaran del 15 al 30

In [39]:
```
df_2 = df[15:30]
df_2
```

Out[39]:

	OrderDate	Region	Rep	Item	Units	UnitCost	Total
15	9/18/2020	East	Jones	Pen Set	16	15.99	255.84
16	10/5/2020	Central	Morgan	Binder	28	8.99	251.72
17	10/22/2020	East	Jones	Pen	64	8.99	575.36
18	11/8/2020	East	Parent	Pen	15	19.99	299.85
19	11/25/2020	Central	Kivell	Pen Set	96	4.99	479.04
20	12/12/2020	Central	Smith	Pencil	67	1.29	86.43
21	12/29/2020	East	Parent	Pen Set	74	15.99	1183.26
22	1/15/2021	Central	Gill	Binder	46	8.99	413.54
23	2/1/2021	Central	Smith	Binder	87	15.00	1305.00
24	2/18/2021	East	Jones	Binder	4	4.99	19.96
25	3/7/2021	West	Sorvino	Binder	7	19.99	139.93
26	3/24/2021	Central	Jardine	Pen Set	50	4.99	249.50
27	4/10/2021	Central	Andrews	Pencil	66	1.99	131.34
28	4/27/2021	East	Howard	Pen	96	4.99	479.04
29	5/14/2021	Central	Gill	Pencil	53	1.29	68.37

Despues vamos a seleccionar los datos estadisticos de Unidad, Total y costo de unidad

In [40]:
```
print ("Estadisticos")
print ("-----------------------------")
df_2[['Units','Total','UnitCost']].describe()
```

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

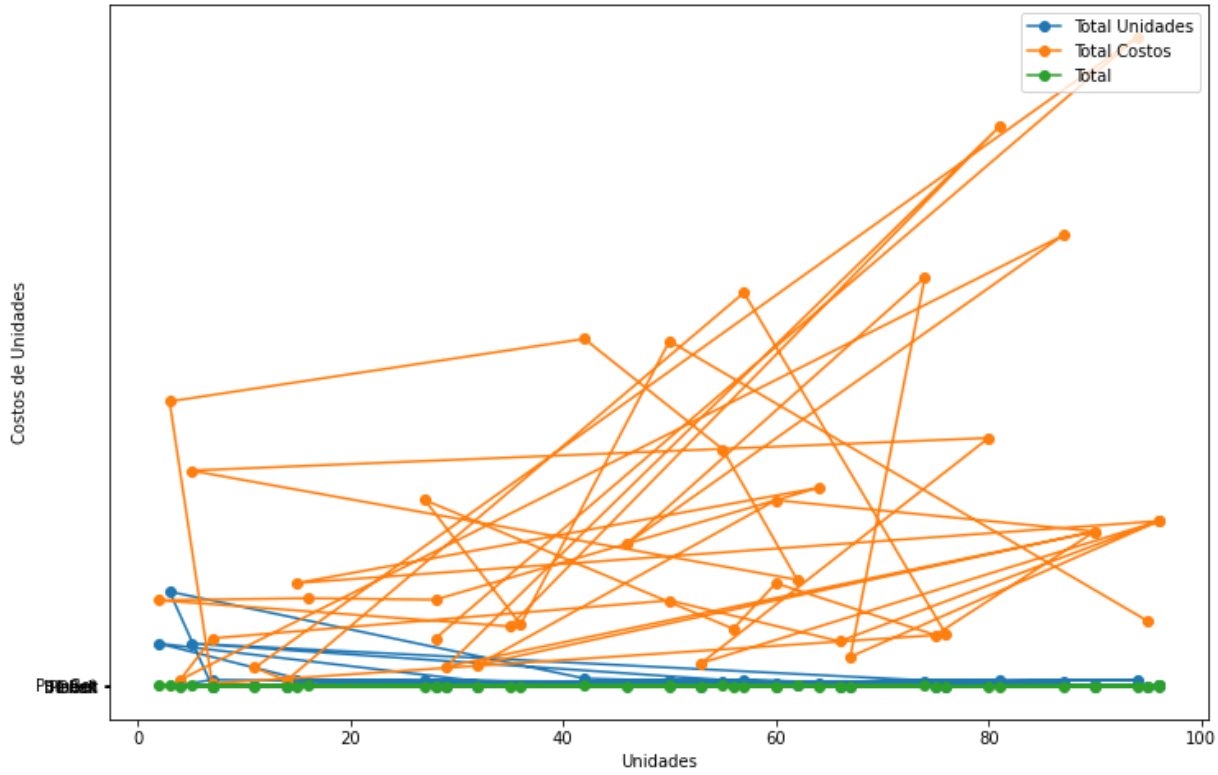
Out[40]:

	Units	Total	UnitCost
count	15.000000	15.000000	15.000000
mean	51.266667	395.878667	9.230667
std	31.337943	382.070907	6.583699
min	4.000000	19.960000	1.290000
25%	22.000000	135.635000	4.990000
50%	53.000000	255.840000	8.990000
75%	70.500000	479.040000	15.495000
max	96.000000	1305.000000	19.990000

Ahora vamos realizar una nueva grafica con algunos puntos de acuerdo a los datos pedidos para la ejecucion

In [58]:
```
#x = np.arange(61)
x = df["Units"]
t1 = df["UnitCost"]
t2 = df["Total"]
t3 = df["Item"]
plt.figure(figsize=(12,8))
plt.plot(x,t1,x,t2,x,t3,marker='o')
plt.xlabel('Unidades')
plt.ylabel('Costos de Unidades')
plt.legend(('Total Unidades','Total Costos', 'Total'), prop = {'size':10},loc='up
```

Out[58]: <matplotlib.legend.Legend at 0x1de27822640>



Continuando con al ejecucion vamos a realizar tres graficas de las tres columnas que hemos utilizado el primer dato ingresado es el rango debe ser su umeracion exacta ya que sino no va a funcionar, luego sigue la informacion del primer dato incluyendo la r de red Mas adelante tenemos el siguiente dato con su respectiva infromacion y la b de blue Por ultimo tenemos el dato del total con g de green

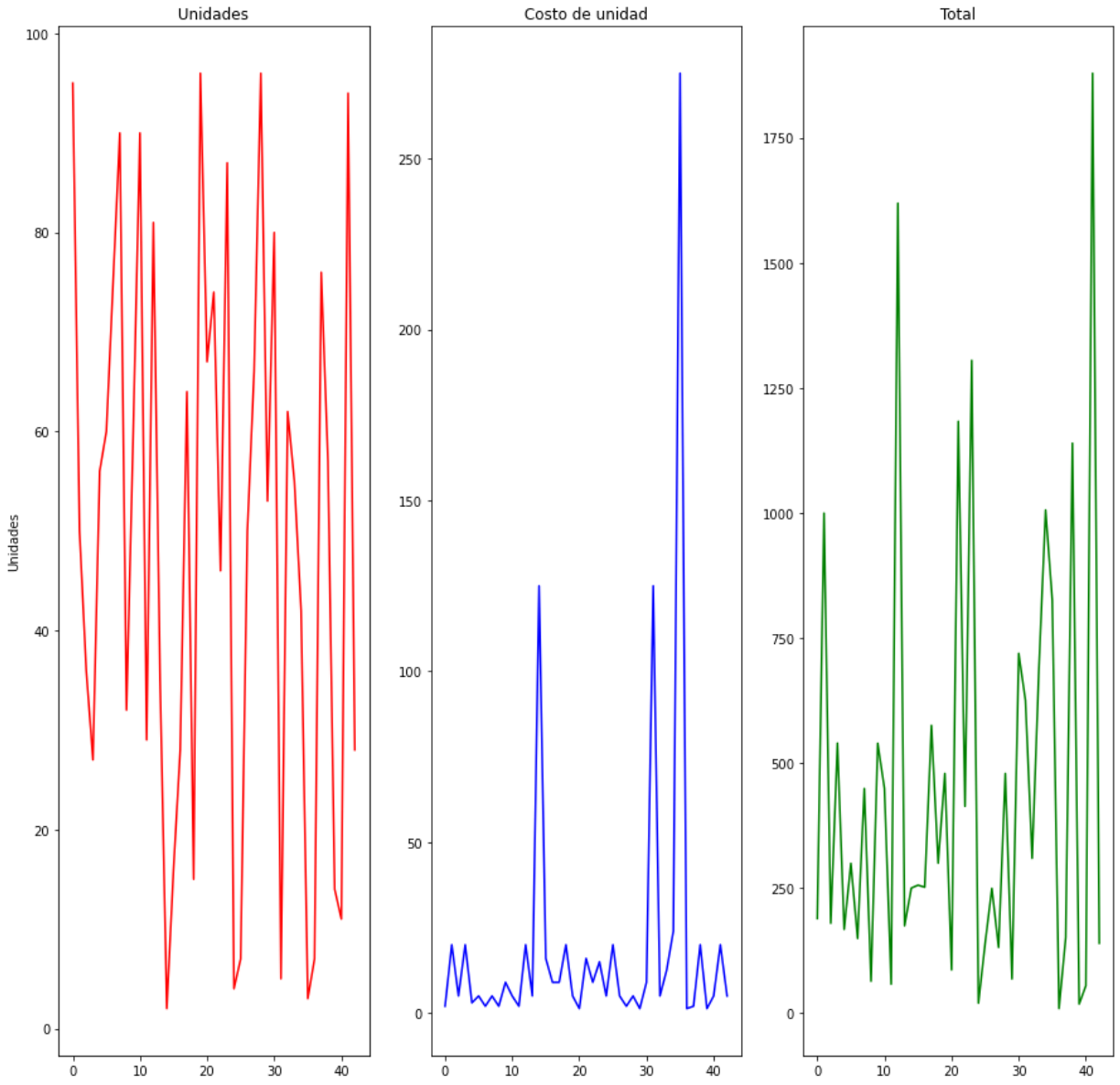
In [76]:
```
x = range(43)
plt.figure(figsize=(15,15))
plt.subplot(131)

t1 = df["Units"]
p1 = plt.plot(x,t1,'r')
plt.ylabel('Unidades')
plt.title(' Unidades ')
plt.subplot(132)

t2 = df["UnitCost"]
p1 = plt.plot(x,t2,'b')
plt.title(' Costo de unidad ')
plt.subplot(133)

t3 = df["Total"]
p1 = plt.plot(x,t3,'g')
plt.title(' Total ')
```

Out[76]: Text(0.5, 1.0, ' Total ')



Despues, vamos a obtener los datos unicos para eso usamos unique y guardamos los dtaos en una variable a la cual se le ha denominado dfclases

In [77]:
```
# OBTENER LOS DATOS UNICOS DE LA TABLA
lis = df["Units"].unique()
lis
dfclases=pd.DataFrame(lis,columns=["Units"])
dfclases
```

Out[77]:

	Units
0	95
1	50
2	36
3	27
4	56
5	60
6	75
7	90
8	32
9	29
10	81
11	35
12	2
13	16
14	28
15	64
16	15
17	96
18	67
19	74
20	46
21	87
22	4
23	7
24	66
25	53
26	80
27	5
28	62
29	55
30	42
31	3

	Units
32	76
33	57
34	14
35	11
36	94

Tabla de Frecuencias Absolutas

En esta parte se obtiene frecuencias absolutas de las unidades y a la columna le ponemos como nombre una J

In [83]:
```
#TABLA DE FRECUENCIAS ABSOLUTAS
# OBTENER FRECUENCIAS ABSOLUTAS DE CADA CLASE
datafi=pd.crosstab(index=df["Units"], columns = "J")
# Creamos una lista con los valores de las frecuencias
li = datafi.values
# agregamos una columna al dataframe
dfclases["J"] = li
#observamos dfclase
dfclases
```

Out[83]:

	Units	J
0	95	1
1	50	1
2	36	1
3	27	1
4	56	2
5	60	1
6	75	1
7	90	1
8	32	1
9	29	1
10	81	2
11	35	1
12	2	1
13	16	1
14	28	1
15	64	1
16	15	1
17	96	2
18	67	1
19	74	1
20	46	1
21	87	1
22	4	2
23	7	1
24	66	1
25	53	1
26	80	1
27	5	1
28	62	1

	Units	J
29	55	1
30	42	1
31	3	1
32	76	1
33	57	2
34	14	1
35	11	1
36	94	2

Frecuencia Relativa

Ahora vamos a sacar la frecuencia relativa y la agregamos a una nueva columna llamada hi

In [84]:
```
# Columna de Frecuencia relativa
total = dfclases.sum(axis=0)
datahi = dfclases["J"]/total["J"] # aqui calculamos la frecuencia
datahi.values
# agregamos nueva columna de frecuencia relativa
dfclases["hi"] = datahi
dfclases
```

Out[84]:

	Units	J	hi
0	95	1	0.023256
1	50	1	0.023256
2	36	1	0.023256
3	27	1	0.023256
4	56	2	0.046512
5	60	1	0.023256
6	75	1	0.023256
7	90	1	0.023256
8	32	1	0.023256
9	29	1	0.023256
10	81	2	0.046512
11	35	1	0.023256
12	2	1	0.023256
13	16	1	0.023256
14	28	1	0.023256
15	64	1	0.023256
16	15	1	0.023256
17	96	2	0.046512
18	67	1	0.023256
19	74	1	0.023256
20	46	1	0.023256
21	87	1	0.023256
22	4	2	0.046512
23	7	1	0.023256
24	66	1	0.023256
25	53	1	0.023256
26	80	1	0.023256
27	5	1	0.023256
28	62	1	0.023256
29	55	1	0.023256
30	42	1	0.023256

	Units	J	hi
31	3	1	0.023256
32	76	1	0.023256
33	57	2	0.046512
34	14	1	0.023256
35	11	1	0.023256
36	94	2	0.046512

Mostraremos el total de dfclases la variable utilizada anteriormente

In [85]:
```
total1 = dfclases.sum(axis=0)
total1
```

Out[85]:
```
Units    1790.0
J          43.0
hi          1.0
dtype: float64
```

Ahora se va a sumar las frecuencias relativas y de la cual se recurrira a calcular la frecuencia absoluta y guardamos el resultado en dfclases

In [86]:
```
# la suma de las frecuencias Relativas nos da 1
# aqui vamos a calcular la frecuencia absoluta
FA = dfclases["J"].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[86]:

	Units	J	hi	FA	HI
0	95	1	0.023256	1	0.023256
1	50	1	0.023256	2	0.046512
2	36	1	0.023256	3	0.069767
3	27	1	0.023256	4	0.093023
4	56	2	0.046512	6	0.139535
5	60	1	0.023256	7	0.162791
6	75	1	0.023256	8	0.186047
7	90	1	0.023256	9	0.209302
8	32	1	0.023256	10	0.232558
9	29	1	0.023256	11	0.255814
10	81	2	0.046512	13	0.302326
11	35	1	0.023256	14	0.325581
12	2	1	0.023256	15	0.348837
13	16	1	0.023256	16	0.372093
14	28	1	0.023256	17	0.395349
15	64	1	0.023256	18	0.418605
16	15	1	0.023256	19	0.441860
17	96	2	0.046512	21	0.488372
18	67	1	0.023256	22	0.511628
19	74	1	0.023256	23	0.534884
20	46	1	0.023256	24	0.558140
21	87	1	0.023256	25	0.581395
22	4	2	0.046512	27	0.627907

	Units	J	hi	FA	HI
23	7	1	0.023256	28	0.651163
24	66	1	0.023256	29	0.674419
25	53	1	0.023256	30	0.697674
26	80	1	0.023256	31	0.720930
27	5	1	0.023256	32	0.744186
28	62	1	0.023256	33	0.767442
29	55	1	0.023256	34	0.790698
30	42	1	0.023256	35	0.813953
31	3	1	0.023256	36	0.837209
32	76	1	0.023256	37	0.860465
33	57	2	0.046512	39	0.906977
34	14	1	0.023256	40	0.930233
35	11	1	0.023256	41	0.953488
36	94	2	0.046512	43	1.000000

Mostramos el total de los datos utilizando una nueva variable.

In [87]:
```
total1 = dfclases.sum(axis=0)
total1
```

Out[87]:
```
Units    1790.000000
J          43.000000
hi          1.000000
FA        803.000000
HI         18.674419
dtype: float64
```

Mostraremos la informacion de los datos

In [89]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 43 entries, 0 to 42
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   OrderDate  43 non-null     object 
 1   Region     43 non-null     object 
 2   Rep        43 non-null     object 
 3   Item       43 non-null     object 
 4   Units      43 non-null     int64  
 5   UnitCost   43 non-null     float64
 6   Total      43 non-null     float64
dtypes: float64(2), int64(1), object(4)
memory usage: 2.5+ KB
```

Ahora vamos a mostrar el costo unitario mas 100

In [91]:
```
df["UnitCost"]+100
```

Out[91]:
```
0     101.99
1     119.99
2     104.99
3     119.99
4     102.99
5     104.99
6     101.99
7     104.99
8     101.99
9     108.99
10    104.99
11    101.99
12    119.99
13    104.99
14    225.00
15    115.99
16    108.99
17    108.99
18    119.99
19    104.99
20    101.29
21    115.99
22    108.99
23    115.00
24    104.99
25    119.99
26    104.99
27    101.99
28    104.99
29    101.29
30    108.99
31    225.00
32    104.99
33    112.49
34    123.95
35    375.00
36    101.29
37    101.99
38    119.99
39    101.29
40    104.99
41    119.99
42    104.99
Name: UnitCost, dtype: float64
```

In [ ]:
```

```