Prueba 2

Cuadrados medios

In [44]:
```
#Librerias
import collections
import math
import matplotlib.pyplot as plt
import pandas as pd
```

In [64]:
```
def valuesDiv1(long,numeroOriginal,digitos):
    numero=int(math.floor((long/2)-(digitos/2)))
    sNumero=str(numeroOriginal)
    numUI=sNumero[numero-2:numero+5]
    return int(numUI)
def valuesDiv(long,numeroOriginal,digitos):
    numero=int(math.floor((long/2)-(digitos/2)))
    sNumero=str(numeroOriginal)
    numUI=sNumero[numero-1:numero+6]
    return int(numUI)
def generarNumeroPseudoaleatorios(iteraciones,semilla,digitos):
    iteraciones=iteraciones
    Xo=semilla
    digitos=digitos
    Xn=0
    XnPo=0
    longuitud=0
    Ui=0
    Rn=0
    arregloRn=[]
    print('Valores')
    for i in range(iteraciones):
        if (i==0):
            Xn=Xo
            XnPo=Xn**2
            longuitud=len(str(XnPo))
            Ui=valuesDiv1(longuitud,XnPo,digitos)
            Rn=Ui/10000000
            Rn=round(Rn, 2)
            arregloRn.append(Rn)
            print(Rn)
        else:
            Xn=Ui
            XnPo=Xn**2
            longuitud=len(str(XnPo))
            Ui=valuesDiv(longuitud,XnPo,digitos)
            Rn=Ui/10000000
            Rn=round(Rn, 2)
            arregloRn.append(Rn)
            print(Rn)
    print('Frecuencia')
    counter=collections.Counter(arregloRn)
    print(counter)
    return arregloRn
iteraciones=100
semilla=74731897457
digitos=7
arregloRn = generarNumeroPseudoaleatorios(iteraciones,semilla,digitos)
```

```
0.85
0.53
0.35
0.46
0.35
0.96
0.16
0.05
0.5
0.26
0.46
0.18
0.3
```

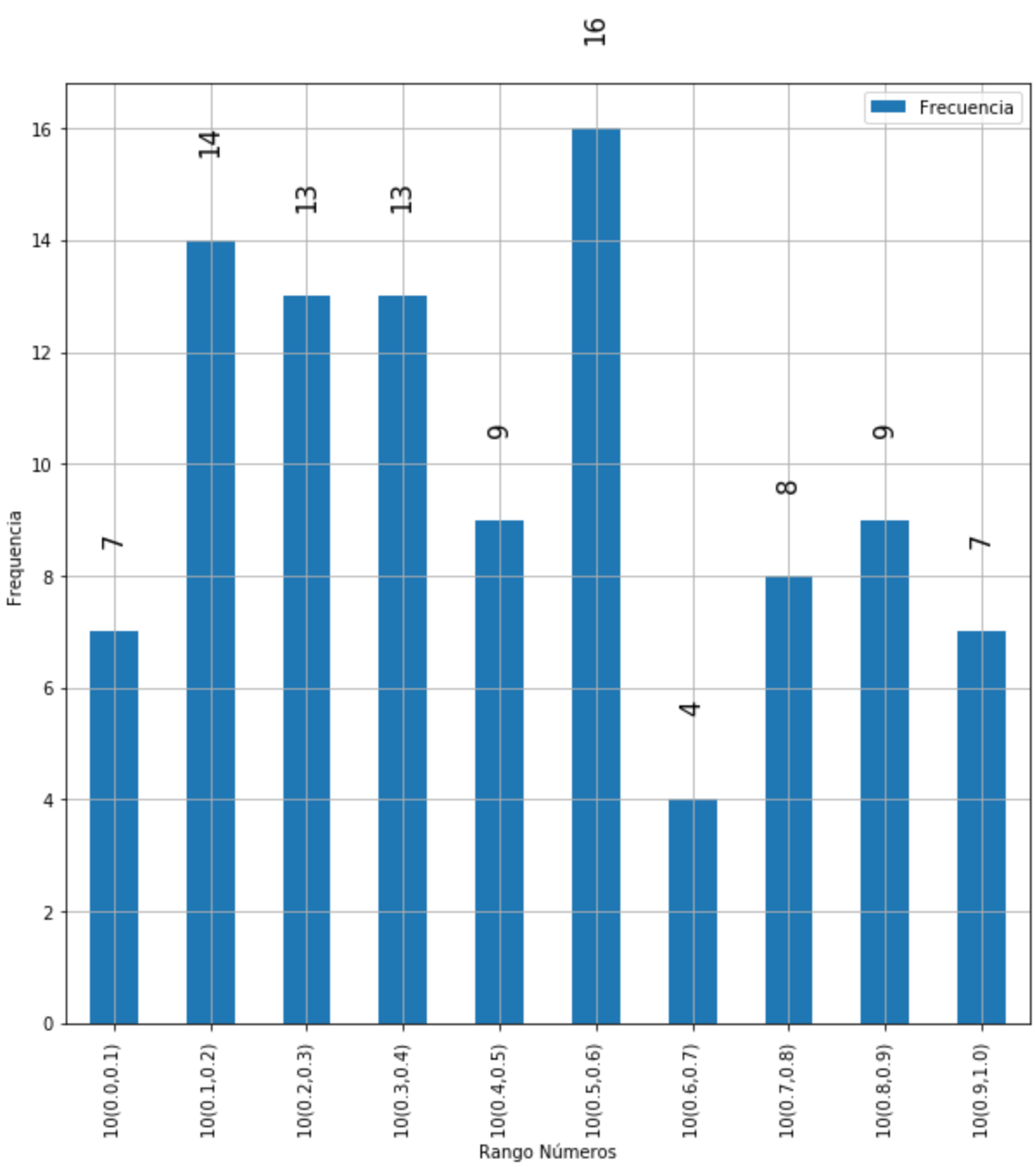
0.7
Frecuencia
Counter({0.13: 5, 0.56: 4, 0.53: 3, 0.74: 3, 0.27: 3, 0.46: 3, 0.98: 3, 0.57: 3, 0.05: 3, 0.29: 3, 0.58: 2, 0.3: 2, 0.12: 2, 0.43: 2, 0.32: 2, 0.82: 2, 0.66: 2, 0.31: 2, 0.87: 2, 0.52: 2, 0.38: 2, 0.9: 2, 0.19: 2, 0.51: 2, 0.35: 2, 0.92: 1, 0.15: 1, 0.06: 1, 0.02: 1, 0.8: 1, 0.25: 1, 0.78: 1, 0.4: 1, 0.48: 1, 0.33: 1, 0.69: 1, 0.2: 1, 0.39: 1, 0.17: 1, 0.45: 1, 0.01: 1, 0.1: 1, 0.77: 1, 0.99: 1, 0.47: 1, 0.83: 1, 0.28: 1, 0.86:

In [70]:

```
plt.rcParams['figure.figsize'] = [10, 10]
rango=[]
frecuencia = []
for intervalo, rangos in enumerate (diccionario.items()):
    valStr = str(n)+"("+rangos[0]+") "
    rango.append(valStr)
    valor= len(rangos[1])
    frecuencia.append(valor)
tupla1=tuple(rango)
tupla2=tuple(frecuencia)
data = {"Rango": tupla1,
        "Frecuencia": tupla2,
        "Repeticion": tupla2}

frecuencia = pd.DataFrame(data)
ax = frecuencia.plot.bar("Rango", "Frecuencia")

for i, bar in enumerate(ax.patches):
    ax.text(bar.get_x() + bar.get_width() / 2, bar.get_height() + 1.5,
            f"{frecuencia['Frecuencia'][i]}",
            horizontalalignment= 'center', verticalalignment='bottom',
            fontsize=15, rotation=90)
ax.set_xlabel("Rango Números")
ax.set_ylabel("Frequencia")
ax.grid(True)
plt.show()
```



CONGRUENCIA LINEAL

In [73]:
```
def nextSeed(multiplicador,xoAnterior,incremento,modulo):
    numero=multiplicador*xoAnterior+incremento
    numXn=numero % modulo
    return numXn
def generarNumeroPseudoaleatorios2(multiplicador,incremento,modulo,iteraciones,semilla):
    a=multiplicador
    b=incremento
    m=modulo
    iteracion=iteraciones
    Xo=semilla
    Xn=0
    Un=0
    arregloRn2=[]
    print('Valores')
    for i in range(iteracion):
        if (i==0):
            Xn=Xo
        else:
            Xn=nextSeed(a,Xn,b,m)
            Xn=Xn
            Un=Xn/m
            Un=round(Un,2)
            arregloRn2.append(Un)
            print(Un)
    print('FRECUENCIA')
    counter=collections.Counter(arregloRn2)
    print(counter)
    return arregloRn2
multiplicador=74731897457
incremento=37747318974
modulo=19
iteraciones=100
semilla=7
arregloRn2 = generarNumeroPseudoaleatorios2(multiplicador,incremento,modulo,iteraciones,ser
```

```
Valores
0.89
0.84
0.95
0.74
0.16
0.32
0.0
0.63
0.37
0.89
0.84
0.95
0.74
0.16
0.32
0.0
0.63
0.37
0.89
0.84
0.95
0.74
0.16
0.32
0.0
```

0.63
0.37
0.89
0.84
0.95
0.74
0.16
0.32
0.0
0.63
0.37
0.89
0.84
0.95
0.74
0.16
0.32
0.0
0.63
0.37
0.89
0.84
0.95
0.74
0.16
0.32
0.0
0.63
0.37
0.89
0.84
0.95
0.74
0.16
0.32
0.0
0.63
0.37
0.89
0.84
0.95
0.74
0.16
0.32
0.0
0.63
0.37
0.89
0.84
0.95
0.74
0.16
0.32
0.0
0.63
0.37
0.89
0.84
0.95
0.74
0.16
0.32

```
0.0
0.63
0.37
0.89
0.84
0.95
0.74
0.16
0.32
0.0
0.63
0.37
FRECUENCIA
Counter({0.89: 11, 0.84: 11, 0.95: 11, 0.74: 11, 0.16: 11, 0.32: 11, 0.0: 11, 0.63: 11,
0.37: 11})
```

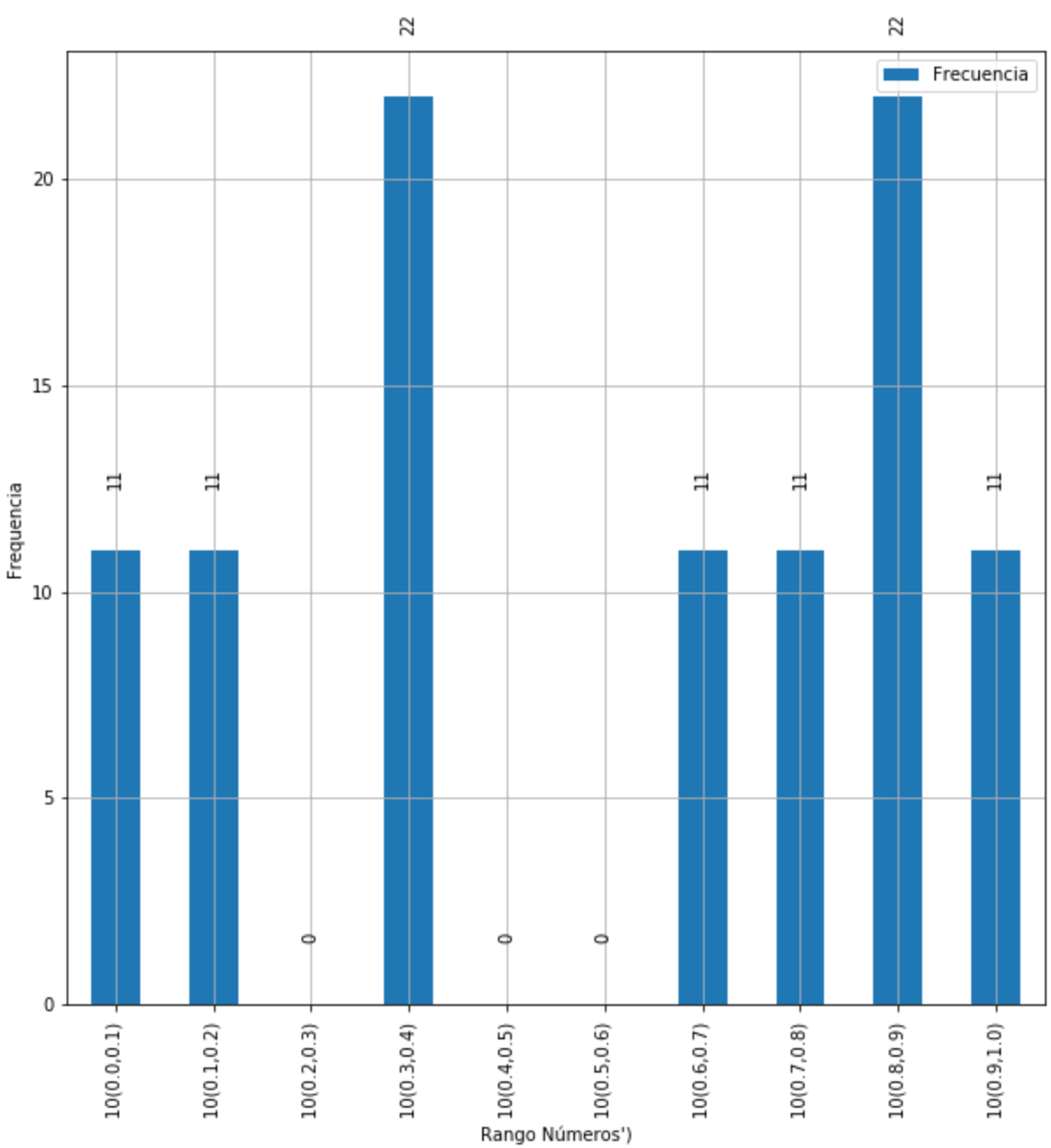
In [75]:
```
def clasificarNumeros2(n,arregloRn):
    grupos = []
    inicio=0.00
    a=0
    b=1
    ranNumeros= {}
    for i in range(n+1):
        grupos.append(round(inicio,2))
        inicio=inicio+(1/n)
    for i in range(len(grupos)-1):
        valInferior=grupos[a]
        valSuperior=grupos[b]
        ranNumeros.update({str(valInferior)+","+str(valSuperior):[]})
        for i in arregloRn:
            if i==0.00:
                if i>=valInferior and i<= valSuperior:
                    ranNumeros[str(valInferior)+","+str(valSuperior)].append(i)
            else:
                if i>valInferior and i<= valSuperior:
                    ranNumeros[str(valInferior)+","+str(valSuperior)].append(i)
        a=b
        b=a+1
    return ranNumeros
diccionario2 = clasificarNumeros(n,arregloRn2)
sumaOi1=0.00
print("Intervalo " , "        Ei","     Oi", "     (Oi-Ei)^2/Ei")
for intervalo, rangos in enumerate (diccionario2.items()):
    porcentajeOi= ((len(rangos[1])-n)**2)/n
    numRepeticion= len(rangos[1])
    sumaOi1+=porcentajeOi
    print(intervalo+1,"          ",str(n)+"("+rangos[0]+") ",  numRepeticion,"        ",por
print('Valor Chi-Cuadrado',sumaOi1)
if sumaOi1<=16.9:
    print('La Diferencia entre la distribución de la muestra y la distribución uniforme es
else:
    print('La Diferencia entre la distribución de la muestra y la distribución uniforme no
```

```
Intervalo          Ei      Oi      (Oi-Ei)^2/Ei
1            10(0.0,0.1)  11          0.1
2            10(0.1,0.2)  11          0.1
3            10(0.2,0.3)  0          10.0
4            10(0.3,0.4)  22          14.4
5            10(0.4,0.5)  0          10.0
6            10(0.5,0.6)  0          10.0
7            10(0.6,0.7)  11          0.1
8            10(0.7,0.8)  11          0.1
9            10(0.8,0.9)  22          14.4
10            10(0.9,1.0)  11          0.1
Valor Chi-Cuadrado 59.300000000000004
La Diferencia entre la distribución de la muestra y la distribución uniforme no es valida
```

In [76]:

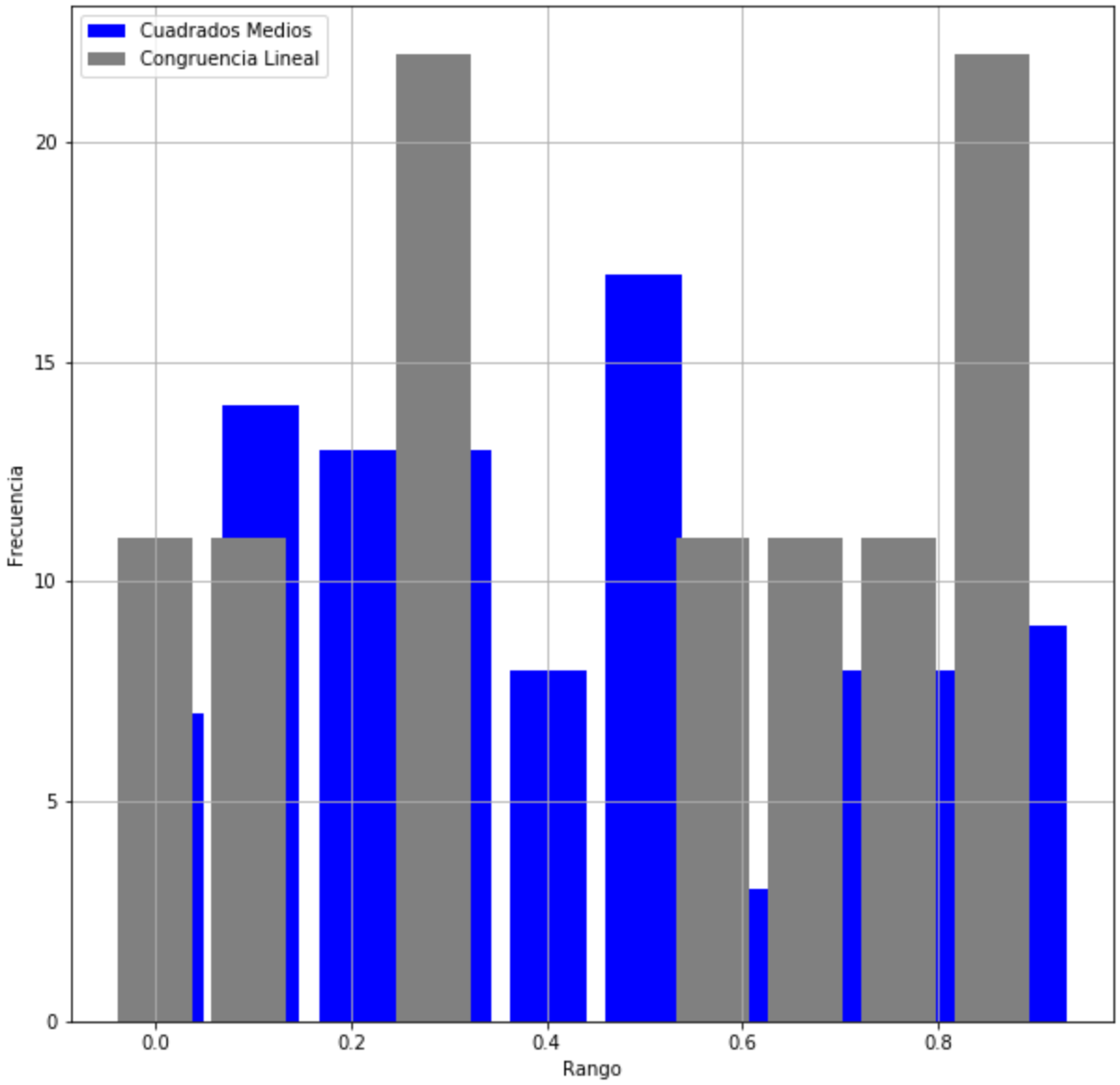
```
rango=[]
frecuencia = []
for intervalo, rangos in enumerate (diccionario2.items()):
    valStr = str(n)+"("+rangos[0]+") "
    rango.append(valStr)
    valor= len(rangos[1])
    frecuencia.append(valor)

tupla1=tuple(rango)
tupla2=tuple(frecuencia)
data = {"Rango": tupla1,
        "Frecuencia": tupla2,
        "Repeticion": tupla2}
frecuencia = pd.DataFrame(data)
ax = frecuencia.plot.bar("Rango", "Frecuencia")
for i, bar in enumerate(ax.patches):
    ax.text(bar.get_x() + bar.get_width() / 2, bar.get_height() + 1.5,
            f"{frecuencia['Frecuencia'][i]}",
            horizontalalignment= 'center', verticalalignment='bottom',
            fontsize=10, rotation=90)
ax.set_xlabel("Rango Números')")
ax.set_ylabel("Frequencia")
ax.grid(True)
plt.show()
```



COMPARACION DE HISTOGRAMAS.

In [81]:
```
plt.rcParams['figure.figsize'] = [10, 10]
plt.hist(arregloRn,color='b', rwidth=0.80,align='left',label='Cuadrados Medios')
plt.hist(arregloRn2,color='grey', rwidth=0.80,align='left',label='Congruencia Lineal')
plt.xlabel('Rango')
plt.ylabel('Frecuencia')
plt.legend(loc='upper left')
plt.grid(True)
plt.show()
print('Valor Cuadrados Medios:',sumaOi)
print('Valor Congruencia Lineal:',sumaOi1)
if (sumaOi<sumaOi1):
    print('VALIDA CUADRADOS MEDIOS')
else:
    print('VALIDA CONGRUENCIA LINEAL')
```



Valor Cuadrados Medios: 13.0
Valor Congruencia Lineal: 59.300000000000004
VALIDA CUADRADOS MEDIOS