Prueba

Nombre: William Paredes

Se buscar encontrar la eficiencia de la generación de números pseudo-aletorios a través de los métodos de cuadrados medios y congruencia lineal, para ello se debe seguir el siguiente proceso:

1. A traves de la misma api generar una semilla diferente.
2. Encontrar el numero de iteraciones hasta que se repita uno de sus datos.
3. Generar 100 simulaciones con diferentes semillas.
4. Generar un histograma con el resultado obtenidos por cada método.
5. Agregar sus conclusiones, opiniones y recomendaciones

Se debe generar un cuaderno de python para la simulación y subir dentro de este apartado.

Metodo Cuadrados Medios

In [91]:

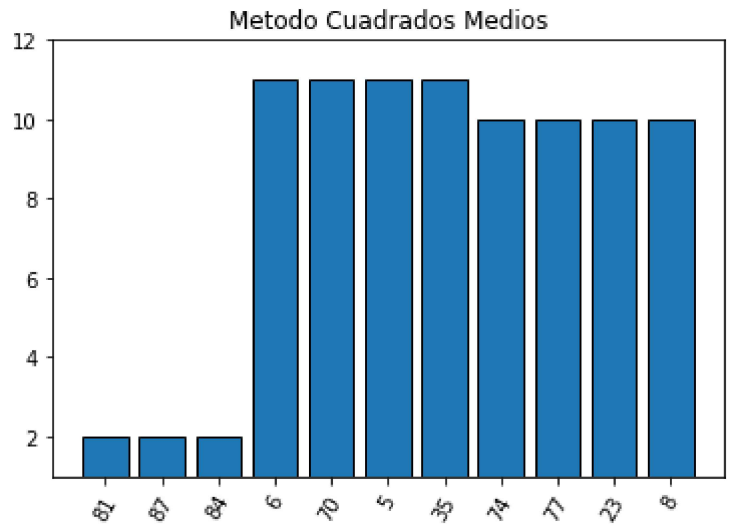
```
Semilla=43242
Digitos=4
nu="1"
iteraciones=[]
for i in range(0,Digitos):
    nu=nu+"0"
for j in range(0,100):
    i=1
    xn=[]
    val= True
    while val:
        semilla3=Semilla
        Semilla2=semilla3*semilla3
        palabra=str(Semilla2)
        longitud=len(palabra)
        para1=(int((longitud/2))-int((Digitos/2)))
        para2=(int((longitud/2))+int((Digitos/2)))
        Res=palabra[int(para1):int(para2)]
        aleatorio=float(Res)/float(nu)
        Semilla=int(Res)
        if (i==1):
            xn.append(Res)
            i=i+1
        else:
            for x in range(0,len(xn)):
                if (xn[x]==Res):
                    iteraciones.append(i)
                    Semilla=int(xn[0])+100-5
                    val=False
            xn.append(Res)
            i=i+1
print(iteraciones)
```

```
[81, 43, 81, 72, 25, 17, 78, 87, 87, 84, 30, 104, 28, 84, 73, 15, 6, 70,
5, 35, 74, 77, 23, 8, 6, 70, 5, 35, 74, 77, 23, 8, 6, 70, 5, 35, 74, 77, 2
3, 8, 6, 70, 5, 35, 74, 77, 23, 8, 6, 70, 5, 35, 74, 77, 23, 8, 6, 70, 5,
35, 74, 77, 23, 8, 6, 70, 5, 35, 74, 77, 23, 8, 6, 70, 5, 35, 74, 77, 23,
8, 6, 70, 5, 35, 74, 77, 23, 8, 6, 70, 5, 35, 74, 77, 23, 8, 6, 70, 5, 35]
```

In [92]:

```
%matplotlib inline
from matplotlib import pyplot as plt
#print(iteraciones)
aux=[]
repetido = []
unico = []
valores = iteraciones
cont=[]

def frecuencia(lista,valor):

    aux=0
    for i in valor:
        for j in lista:
            if i == j:
                aux=aux+1
        cont.append(aux)
        aux=0
    return cont

for x in valores:
    del aux[:]
    if x not in unico:
        unico.append(x)
    else:
        if x not in repetido:
            repetido.append(x)
fra=frecuencia(valores,repetido)

print(len(repetido))
fechas = repetido
primas = cont
plt.bar(range(len(repetido)), primas, edgecolor='black')
plt.xticks(range(len(repetido)), fechas, rotation=60)
plt.title("Metodo Cuadrados Medios")
plt.ylim(min(primas)-1, max(primas)+1)
plt.show()
```

11



Metodo Congruencia Lineal

In [88]:

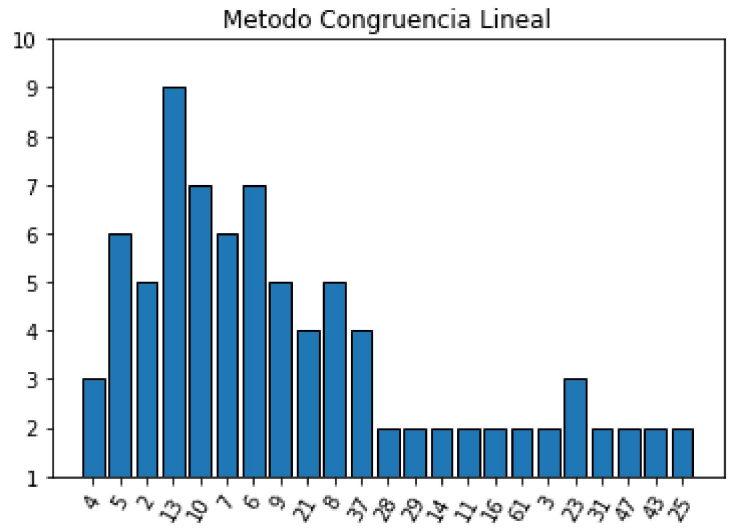
```
a=1
b=1
Semilla=4480
m=3
iteraciones=[]
for j in range(0,100):
    i=1
    ui1=[]
    xn=0
    ui=0.0
    aux=0
    val= True
    xn=Semilla
    while val:
        aux=((a*xn)+b)%(m)
        ui=aux/m
        xn=aux
        if (i==1):
            ui1.append(ui)
            i=i+1
        else:
            for x in range(0,len(ui1)):
                if (ui1[x]==ui):
                    iteraciones.append(i)
                    Semilla=int(xn)
                    a=a+1
                    b=b+3
                    m=m+1
                    val=False
            ui1.append(ui)
            i=i+1
print(iteraciones)
print(len(iteraciones))
```

```
[4, 2, 5, 4, 7, 3, 10, 5, 6, 5, 13, 2, 13, 5, 9, 10, 10, 2, 7, 6, 23, 6, 2
1, 13, 28, 8, 29, 13, 2, 5, 16, 9, 13, 11, 37, 10, 13, 7, 21, 7, 8, 7, 37,
2, 47, 6, 43, 21, 25, 14, 53, 28, 6, 9, 10, 29, 30, 14, 61, 11, 10, 6, 13,
16, 34, 9, 67, 5, 71, 12, 19, 37, 61, 3, 31, 13, 79, 4, 82, 21, 42, 8, 9,
8, 85, 7, 23, 37, 13, 23, 31, 47, 10, 8, 49, 43, 46, 6, 101, 25]
100
```

In [89]:

```
%matplotlib inline
from matplotlib import pyplot as plt
print(iteraciones)
aux=[]
repetido = []
unico = []
valores = iteraciones
cont=[]

def frecuencia(lista,valor):
    aux=0
    for i in valor:
        for j in lista:
            if i == j:
                aux=aux+1
        cont.append(aux)
        aux=0
    return cont

for x in valores:
    del aux[:]
    if x not in unico:
        unico.append(x)
    else:
        if x not in repetido:
            repetido.append(x)

fra=frecuencia(valores,repetido)
print(len(repetido))
fechas = repetido
primas = cont
plt.bar(range(len(repetido)), primas, edgecolor='black')
plt.xticks(range(len(repetido)), fechas, rotation=60)
plt.title("Metodo Congruencia Lineal")
plt.ylim(min(primas)-1, max(primas)+1)
plt.show()
```

```
[4, 2, 5, 4, 7, 3, 10, 5, 6, 5, 13, 2, 13, 5, 9, 10, 10, 2, 7, 6, 23, 6, 2
1, 13, 28, 8, 29, 13, 2, 5, 16, 9, 13, 11, 37, 10, 13, 7, 21, 7, 8, 7, 37,
2, 47, 6, 43, 21, 25, 14, 53, 28, 6, 9, 10, 29, 30, 14, 61, 11, 10, 6, 13,
16, 34, 9, 67, 5, 71, 12, 19, 37, 61, 3, 31, 13, 79, 4, 82, 21, 42, 8, 9,
8, 85, 7, 23, 37, 13, 23, 31, 47, 10, 8, 49, 43, 46, 6, 101, 25]
23
```



Conclusiones

Por lo que podemos apreciar en la diferentes Graficas en el Primer metodo po demos apreciar como no existe muchos coincidencias en las iteraciones mientras q ue el segundo es casi el doble por lo que existe una taza mas alta de numeros re petidos

Opiniones

Con respecto alas pruebas realizadas podemos decir que el metodo mas rentable se ria de Cuadrados medio debido al poco casos de repeticiones en las iteraciones a demas como tiene muchas iteraciones altas

Recomendaciones

Deberiamos usar numeros altos como semillaS
Al momento de manipular a,b y m sumar numeros primos
Hacer varias pruebas para obtener resultados deseados