# 6_Central Limit Theorem

In [40]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [41]:
```
# Generate random data by using list comprehension

pop_data = [np.random.randint(10,100) for i in range(10000)]
#pop_data
```

In [42]:
```
# the above line of code could be written as:
pop_data = []
for i in range(10000):
    pop_data.append(np.random.randint(10,100))
#pop_data
```

In [43]:
```
len(pop_data)
```

Out[43]: 10000

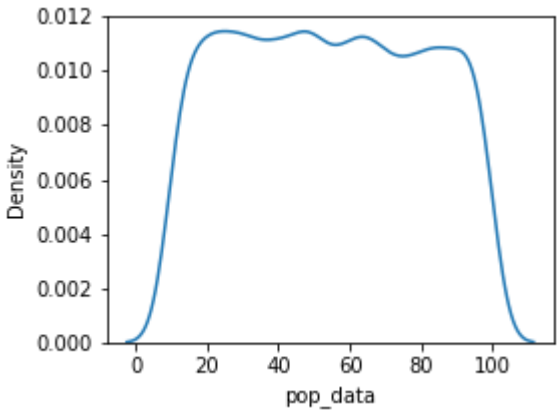
In [44]:
```
# TO convert population data into a csv file
pop_table = pd.DataFrame({'pop_data':pop_data})
pop_table
```

Out[44]:

| | pop_data |
|---|---|
| **0** | 55 |
| **1** | 22 |
| **2** | 87 |
| **3** | 46 |
| **4** | 29 |
| **...** | ... |
| **9995** | 51 |
| **9996** | 60 |
| **9997** | 42 |
| **9998** | 99 |
| **9999** | 48 |

10000 rows × 1 columns

In [45]:
```
plt.figure(figsize=(4,3))
sns.kdeplot(x='pop_data', data=pop_table)
plt.show()
```




### above graph shows that our data is not normally distributed, so we will apply CLT

In [46]:
```
# First we will pick up random samples from population data
# Pre-req: Sample should not be more than 10% population and more than 30 samples s

# so calculate 10% of 10000 data

10/100 * 10000
```

Out[46]:
```
1000.0
```

### That means i.e. n>30 and n< 1000, so are taking n= [50,500]

In [47]:
```
# To pick random data from population data
np.random.choice(pop_data)
```

Out[47]:
```
82
```

In [ ]:
```
# So will take sample data less than 1000
sample_mean = []
# to take number of sample data 50 (to meet requirement n>30)
for no_of_sample in range(50):
    sample_data = []
    # to take number of sample data less than 1000 (so will take 500 sample)
    for i in range(500):
        sample_data.append(np.random.choice(pop_data))
    # To calculate mean of sample data
    sample_mean.append(np.mean(sample_data))
```

In [ ]:
```
len(sample_data), len(sample_mean)
```

In [ ]:
```
sample_data
```

In [ ]:
```
sample_mean
```

In [ ]:
```
# To see data in sample_mean is normally distributed or not
sample_mean_DF = pd.DataFrame({"Sample_mean":sample_mean})
```

In [ ]:
```
sample_mean_DF
```

In [ ]:
```
plt.figure(figsize=(4,5))
sns.kdeplot(x="Sample_mean", data=sample_mean_DF)
plt.show()
```

## So the data is normally distributed

In [ ]:
```
# To meat another requirement of CLT that is the mean of population data and the me
# so we will check the both means
np.mean(pop_data), np.mean(sample_mean)
```