01. Simple Random Sampling: This involves selecting a sample from a population at random, with each individual in the population having an equal chance of being selected.

```
import random

population = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
sample_size = 5

sample = random.sample(population, sample_size)
print(sample)
```

```
[3, 4, 1, 2, 6]
```

02. Stratified Sampling: This involves dividing the population into strata or subgroups based on a characteristic, and then selecting a sample from each stratum.

```
import random

population = {
    'A': [1, 2, 3, 4, 5],
    'B': [6, 7, 8, 9, 10]
}

stratum_sample_size = {
    'A': 2,
    'B': 3
}

sample = []
for stratum in population.keys():
    stratum_sample = random.sample(population[stratum],
stratum_sample_size[stratum])
    sample.extend(stratum_sample)

print(sample)
```

```
[3, 5, 7, 6, 8]
```

03. Cluster Sampling: This involves dividing the population into clusters, selecting a few clusters at random, and then sampling all individuals in the selected clusters.

```
import random

population = {
    'Cluster 1': [1, 2, 3, 4, 5],
    'Cluster 2': [6, 7, 8, 9, 10],
    'Cluster 3': [11, 12, 13, 14, 15]
}

clusters_to_sample = 2

sample = []
selected_clusters = random.sample(population.keys(),
clusters_to_sample)
for cluster in selected_clusters:
    sample.extend(population[cluster])

print(sample)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_9832\2777618067.py:12:
DeprecationWarning: Sampling from a set deprecated
since Python 3.9 and will be removed in a subsequent version.
  selected_clusters = random.sample(population.keys(),
clusters_to_sample)
```

04. Systematic Sampling: This involves selecting every nth individual from a population, where n is a fixed interval.

```
import random

population = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
n = 3

sample = [population[i] for i in range(0, len(population), n)]
print(sample)
```

```
[1, 4, 7, 10]
```

05. Convenience Sampling: This involves selecting individuals who are easily accessible or readily available.

```
import random
```

```
population = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
sample_size = 5

sample = random.choices(population, k=sample_size)
print(sample)
```

```
[5, 7, 7, 5, 10]
```

other methods using different libraries to get the sample

random.sample(): This function in the random module can be used to randomly select a specified number of items from a given sequence.

numpy.random.choice(): This function in the numpy module can be used to randomly select elements from an array or list, with or without replacement.

pandas.DataFrame.sample(): This method in the pandas library can be used to randomly sample rows or columns from a DataFrame.

scipy.stats.rv_discrete.rvs(): This function in the scipy module can be used to randomly generate samples from a discrete distribution.