```
# Preliminaries
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)


from scipy import stats
import numpy as np
```

Data Set

The file `student_score.csv` contains student score data from two schools: `MS` and `GP`. Each student from these schools is given a unique alphanumeric `student_id`. The `address` column either reads `U` or `R` to denote whether the student lives in an urban or a rural area. A count of absent days for each student is recorded under the column `absences`. Entries under columns `subject_1`, `subject_2` and `subject_3` denote the marks scored in three different subjects.

```
# Read the CSV file into a Pandas DataFrame
# https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html
student_scores = pd.read_csv("student_scores.csv", index_col=False)


# How large is the dataset
student_scores.shape
```

```
(649, 9)
```

```
student_scores.loc[student_scores['school']=='MS'].shape
```

```
(226, 9)
```

```
# Display the contents of three randomly sampled rows
student_scores.sample(3)
```

	school	student_id	sex	age	address	absences	subject_1	subject_2	subject_3
501	MS	STD79	M	16	U	8	14	12	13
336	GP	STD337	M	18	U	2	15	16	16
609	MS	STD187	F	18	U	0	11	11	12

```
# Display the type of data in each column
# https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.dtypes.html
student_scores.dtypes
```

```
school        object
student_id    object
sex           object
age            int64
address       object
absences       int64
subject_1      int64
subject_2      int64
subject_3      int64
dtype: object
```

```
# Display just the unique entries under column `school`
student_scores['school'].unique()
```

```
array(['GP', 'MS'], dtype=object)
```

Questions

Q1.A

Construct a sample of size 20 of students from the school `GP` by:

a. sampling with replacement

b. sampling without replacement

Repeat the above for school `MS`.

```
#For school GP

#Sample of size 20 from school GP with replacement
sample_gp_with_replacement=student_scores[student_scores['school']=='GP'].sample(n=20,replace=True)

#Sample of size 20 from school GP without replacement
sample_gp_without_replacement=student_scores[student_scores['school']=='GP'].sample(n=20, replace=False)

#printing
print("Sample of GP with replacement:")
print(sample_gp_with_replacement)


print("\nSample of GP without replacement:")
print(sample_gp_without_replacement)
```

```
Sample of GP with replacement:
    school student_id sex  age address  absences  subject_1  subject_2  \
30      GP      STD31   M   15       U         0         10         11
115     GP     STD116   M   16       U         6         16         14
88      GP      STD89   M   16       U         6         12         10
407     GP     STD408   F   21       U         0          9         12
35      GP      STD36   F   15       U         4         11         11
179     GP     STD180   M   17       U        10          8          7
350     GP     STD351   M   19       R         0          9         10
298     GP     STD299   F   17       U         2         10         11
```

```
393     GP     STD394   F   18       U          4          14          14
52      GP      STD53   M   15       U          4          10           9
322     GP     STD323   F   19       R          0           9           8
105     GP     STD106   F   15       U         10          10          10
336     GP     STD337   M   18       U          2          15          16
130     GP     STD131   F   15       R          0          10          11
57      GP      STD58   M   15       U          8          15          15
380     GP     STD381   F   17       U          0          13          12
155     GP     STD156   M   17       U         22           9           7
255     GP     STD256   F   18       U         14           8           7
366     GP     STD367   F   17       U          0          12          12
124     GP     STD125   F   16       U          0          12          11

     subject_3
30          11
115         14
88          11
407         12
35          11
179          8
350         11
298         12
393         15
52           9
322         10
105         10
336         16
130         12
57          16
380         13
155          6
255          7
366         13
124         11

Sample of GP without replacement:
    school student_id sex  age address  absences  subject_1  subject_2  \
29      GP      STD30   M   16       U          4          12          11
390     GP     STD391   F   18       R          6          14          13
314     GP     STD315   M   17       R          2          16          17
145     GP     STD146   F   16       U          4           9           9
360     GP     STD361   F   18       U          8          11          12
```

```
     415     GP     STD416   F   19       U         5          9         10
     282     GP     STD283   M   18       U         8          7          8
     260     GP     STD261   F   16       U         4         12         11
     337     GP     STD338   F   17       U         0         17         18
     243     GP     STD244   F   17       U         0         15         15
```

```
#For school MS

#Sample of size 20 from school MS with replacement
sample_ms_with_replacement=student_scores[student_scores['school']=='MS'].sample(n=20,replace=True)

#Sample of size 20 from school MS without replacement
sample_ms_without_replacement=student_scores[student_scores['school']=='MS'].sample(n=20, replace=False)

#printing
print("Sample of MS with replacement:")
print(sample_ms_with_replacement)


print("\nSample of MS without replacement:")
print(sample_ms_without_replacement)
```

640	MS	STD218	M	18	R	0	7	7
477	MS	STD55	M	15	U	11	12	10
594	MS	STD172	F	18	U	0	18	18
639	MS	STD217	M	19	R	0	5	8
575	MS	STD153	F	18	R	8	10	11
619	MS	STD197	F	18	U	6	13	12
445	MS	STD23	M	15	R	8	7	9
558	MS	STD136	M	17	R	0	8	13
520	MS	STD98	F	16	U	6	6	8
461	MS	STD39	F	16	R	0	13	12
447	MS	STD25	M	17	R	8	8	10
491	MS	STD69	F	19	U	12	7	8
577	MS	STD155	M	19	R	8	10	9
494	MS	STD72	F	16	R	0	8	9
591	MS	STD169	F	18	U	2	12	13
425	MS	STD3	F	15	R	6	10	10
587	MS	STD165	F	18	R	3	7	6
424	MS	STD2	F	16	R	0	12	12
580	MS	STD158	M	19	R	4	8	9

	subject_3
596	18
640	0
477	11
594	18
639	0
575	10
619	13
445	9
558	10
520	8
461	14
447	9
491	9
577	11
494	9
591	14
425	10
587	8
424	12
580	10

Q1.B

Do you notice any difference between the samples generated with and without replacement for school `GP`? Explain why or why not.

```
Differences Between Samples with and without Replacement for School GP
```

When comparing the samples generated with and without replacement for school GP, we observe that the sample taken **with replacement** contains duplicate entries, means some student IDs appear more than once. For example, the student ID `STD293` appears twice in the sample with replacement. This is because, in sampling with replacement, each student has the same probability of being selected in each draw, and once selected, the student is returned to the pool for potential re-selection.

In opposite, the sample taken **without replacement** contains only unique student IDs, with no duplicates. This is because, once a student is selected, they are removed from the pool of potential candidates, ensuring that each student can only be selected once. This method ensures that all selected students in the sample without replacement are distinct.

Therefore, the main difference between these two sampling methods is that sampling with replacement can result in duplicate selections, while sampling without replacement guarantees unique selections.

```
Start coding or generate with AI.
```

Q2.

Compute the sample mean and sample variance of all three subject marks from the data generated in `Q1.A.b` for both school `GP` and school `MS`. What are these sample means and sample variances estimating?

```
#For GP
gp_means=sample_gp_without_replacement[['subject_1','subject_2','subject_3']].mean()
gp_variances=sample_gp_without_replacement[['subject_1','subject_2','subject_3']].var()

#For MS
ms_means=sample_ms_without_replacement[['subject_1','subject_2','subject_3']].mean()
ms_variances=sample_ms_without_replacement[['subject_1','subject_2','subject_3']].var()

#Display
gp_means, gp_variances, ms_means, ms_variances
```

```
(subject_1    11.90
 subject_2    12.15
 subject_3    12.55
 dtype: float64,
 subject_1    9.252632
 subject_2    8.028947
 subject_3    9.102632
 dtype: float64,
 subject_1     9.90
 subject_2    10.60
 subject_3    10.15
 dtype: float64,
 subject_1    12.410526
 subject_2    10.147368
 subject_3    20.344737
 dtype: float64)
```

`Sample Means` and `Sample Variances`: These statistics are used to estimate the population means and population variances of the marks in the three subjects for students in schools GP and MS.

```
Start coding or generate with AI.
```

Q3.

Using the samples generated in `Q1.A.b` for both schools:

a. Construct a 95% confidence interval of the average marks received by students from school `GP` in `subject_3`. Does the true (population) mean of subject 3 marks of students in school `GP` lie inside the confidence interval you generated?

b. Do the same for school `MS`.

```
#Function to calculate confidence interval
def confidence_interval(data, confidence=0.95):
    mean=np.mean(data)
    sem=stats.sem(data)
    interval=sem*stats.t.ppf((1+confidence)/2.,len(data)-1)
    return mean-interval,mean+interval

#95% CI for GP Subject 3
gp_subject_3_data=sample_gp_without_replacement['subject_3']
gp_subject_3_ci=confidence_interval(gp_subject_3_data)

#95% CI for MS Subject 3
ms_subject_3_data=sample_ms_without_replacement['subject_3']
ms_subject_3_ci=confidence_interval(ms_subject_3_data)

#Display
gp_subject_3_ci, ms_subject_3_ci
```

```
((11.13797396467201, 13.962026035327991),
 (8.039014452185661, 12.26098554781434))
```

`95% Confidence Interval`: This interval estimates the range within which the true population mean of subject 3 marks lie with 95% confidence. We are 95% confident that the true mean of subject_3 marks for students in school `GP` lies between `11.14` and `13.96` and true mean of subject_3 marks for students in school `MS` lies between `8.04` and `12.26`.

Start coding or generate with AI.

Q4.

Suppose you have a coin and you want to test the null hypothesis that the coin is fair. You toss the coin 100 times and observe 70 heads (and 30 tails).

Would you accept or reject the hypothesis? Justify your answer.

`Hypothesis Testing:`

`Null Hypothesis (H0)` : The coin is fair (P(heads) = 0.5).

`Alternative Hypothesis (H1)` : The coin is not fair (P(heads) ≠ 0.5)

```
#Given data
observed_heads=70
total_tosses=100

#Perform a binomial test using binomtest
result=stats.binomtest(observed_heads,total_tosses,p=0.5,alternative='two-sided')

#Extract p-value
p_value=result.pvalue

#Determine whether to accept or reject the null hypothesis
reject_null=p_value < 0.05

#Display
p_value,reject_null
```

```
(7.85013964559367e-05, True)
```

`P-value` : 7.85013964559367e-05

This p-value represents the probability of observing 70 or more heads (or 30 or fewer heads) out of 100 coin tosses, assuming the coin is fair. A p-value this low (approximately 0.0000785) is much smaller than the common significance level of 0.05.

```
Decision: True (Reject the null hypothesis)
```

Since the p-value is less than 0.05, we reject the null hypothesis.

```
Start coding or generate with AI.
```

--------- END ----------