Section C Coding Practices

Import Necessary Libraries

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import linregress

import warnings
warnings.filterwarnings("ignore")
```

Load dataset using Google Drive

```
# Mount Google Drive
from google.colab import drive
drive.mount('/content/drive')

# Load the dataset file and create into a dataframe object
df = pd.read_csv("/content/drive/MyDrive/DAML/Assignment/student-mat.csv")
```

Load dataset in local

```
# Load the dataset file and create into a dataframe object
df = pd.read_csv("student-mat.csv")
```

```
df
```

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	freetime	goout	Dalc	Walc	health	absences	G1	G2	G3
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	3	4	1	1	3	6	5	6	6
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	3	3	1	1	3	4	5	5	6
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	3	2	2	3	3	10	7	8	10
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	2	2	1	1	5	2	15	14	15
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	3	2	1	2	5	4	6	10	10
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
390	MS	M	20	U	LE3	A	2	2	services	services	...	5	5	4	4	5	4	11	9	9	9
391	MS	M	17	U	LE3	T	3	1	services	services	...	2	4	5	3	4	2	3	14	16	16
392	MS	M	21	R	GT3	T	1	1	other	other	...	5	5	3	3	3	3	3	10	8	7
393	MS	M	18	R	LE3	T	3	2	services	other	...	4	4	1	3	4	5	0	11	12	10
394	MS	M	19	U	LE3	T	1	1	other	at_home	...	3	2	3	3	3	5	5	8	9	9

395 rows × 33 columns

Question 1

```
# Print the number of columns in the dataset
print(len(df.columns.tolist()), "\n")

# Print the column names
print(df.columns)
```

```
33

Index(['school', 'sex', 'age', 'address', 'famsize', 'Pstatus', 'Medu', 'Fedu',
       'Mjob', 'Fjob', 'reason', 'guardian', 'traveltime', 'studytime',
       'failures', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery',
       'higher', 'internet', 'romantic', 'famrel', 'freetime', 'goout', 'Dalc',
       'Walc', 'health', 'absences', 'G1', 'G2', 'G3'],
      dtype='object')
```

```
# Print the name, number of non-null values, and dtype of each attribute
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 395 entries, 0 to 394
Data columns (total 33 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   school      395 non-null    object
 1   sex         395 non-null    object
 2   age         395 non-null    int64
 3   address     395 non-null    object
 4   famsize     395 non-null    object
 5   Pstatus     395 non-null    object
 6   Medu        395 non-null    int64
 7   Fedu        395 non-null    int64
 8   Mjob        395 non-null    object
 9   Fjob        395 non-null    object
 10  reason      395 non-null    object
 11  guardian    395 non-null    object
 12  traveltime  395 non-null    int64
 13  studytime   395 non-null    int64
 14  failures    395 non-null    int64
```

```
 15  schoolsup   395 non-null    object
 16  famsup      395 non-null    object
 17  paid        395 non-null    object
 18  activities  395 non-null    object
 19  nursery     395 non-null    object
 20  higher      395 non-null    object
 21  internet    395 non-null    object
 22  romantic    395 non-null    object
 23  famrel      395 non-null    int64
 24  freetime    395 non-null    int64
 25  goout       395 non-null    int64
 26  Dalc        395 non-null    int64
 27  Walc        395 non-null    int64
 28  health      395 non-null    int64
 29  absences    395 non-null    int64
 30  G1          395 non-null    int64
 31  G2          395 non-null    int64
 32  G3          395 non-null    int64
dtypes: int64(16), object(17)
memory usage: 102.0+ KB
```

Question 2

```
# Determine the total number of attributes within the dataset
print(f"Total number of attributes within the dataset is {len(df.columns)}")
```

```
Total number of attributes within the dataset is 33
```

Question 3

```
# Assess the dataset's dimensions to identify both the number of rows and columns
print(f"The dataset has {df.shape[0]} rows with {df.shape[1]} columns")
```

```
The dataset has 395 rows with 33 columns
```

Question 4

```
# Calculate the average values for "Dalc," "Walc," and "days of absences,"
# rounding these figures to two decimal places for precision
Dalc_avg = round(df["Dalc"].mean(), 2)
Walc_avg = round(df["Walc"].mean(), 2)
doa_avg = round(df["absences"].mean(), 2)

print(f"The average of Dalc is: {Dalc_avg}")
print(f"The average of Walc is: {Walc_avg}")
print(f"The average of days of absences is: {doa_avg}")
```

```
The average of Dalc is: 1.48
The average of Walc is: 2.29
The average of days of absences is: 5.71
```

Question 5

```
# Finding both the minimum and maximum values of "days of absences"
doa_min = df["absences"].min()
doa_max = df["absences"].max()

print(f"The minimum value of days of absences is {doa_min}")
print(f"The maximum value of days of absences is {doa_max}")
```

```
The minimum value of days of absences is 0
The maximum value of days of absences is 75
```

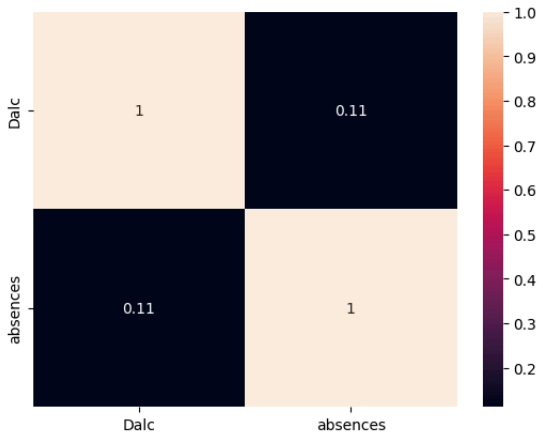
Question 6

```
# Calculate the correlation between these two attributes to quantify their relationship
dtoa_corr = df["Dalc"].corr(df["absences"])
atod_corr = df["absences"].corr(df["Dalc"])

print(f"The correlation between Dalc and days of absences is {round(dtoa_corr, 4)}")
print(f"The correlation between days of absences and Dalc is {round(atod_corr, 4)}")
print()

# Visualize this correlation using a heatmap
sns.heatmap(df[["Dalc", "absences"]].corr(), annot = True)
plt.show()
```

```
The correlation between Dalc and days of absences is 0.1119
The correlation between days of absences and Dalc is 0.1119
```



Question 7

Question (a)

```
# The range of days for absences observed in the dataset
doa_min = df["absences"].min()
doa_max = df["absences"].max()

print(f"The minimum value of days of absences is {doa_min}.")
print(f"The maximum value of days of absences is {doa_max}.")
print(f"Therefore, the days of absences ranges from {doa_min} to {doa_max}, gap is {doa_max - doa_min}.")
```

```
The minimum value of days of absences is 0.
The maximum value of days of absences is 75.
Therefore, the days of absences ranges from 0 to 75, gap is 75.
```

Question (b)

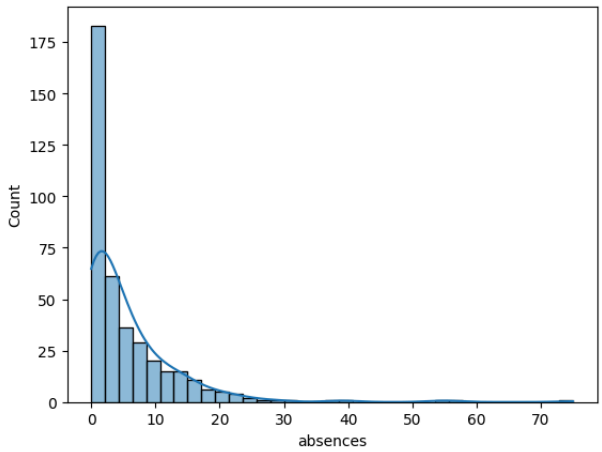
```
# Identify the most and least frequent days for absences among students
doa_freqs = df["absences"].value_counts()

print(f"The most frequent days of absences is {doa_freqs.idxmax()} with frequency of {doa_freqs.max()}.")
print(f"The most frequent days of absences is {doa_freqs.idxmin()} with frequency of {doa_freqs.min()}.")
```

```
The most frequent days of absences is 0 with frequency of 115.
The most frequent days of absences is 38 with frequency of 1.
```

Question (c)

```
# Create a histogram to visualize the distribution of days for absences
sns.histplot(df["absences"], kde = True)
plt.show()
```



Question (d)

```
# Apply linear regression function from scipy into the dataset to see the relationship
slope, intercept, r_value, p_value, std_err = linregress(df["Dalc"], df["absences"])
pred = intercept + slope * df['Dalc']

# Calculate the R-squared value to observe the experiment performance
r2_score = round(r_value ** 2, 4)

print(f"R-squared evaluation score is: {r2_score}")
```

```
R-squared evaluation score is: 0.0125
```

```
# Illustration of plots to represent the relationship between "Dalc" and "absences"
plt.figure(figsize=(8, 6))
plt.scatter(df['Dalc'], df['absences'], color='blue')
plt.plot(df['Dalc'], pred, color='red')
plt.title('Scatterplot between Dalc and Number of days of absences')
plt.xlabel('Dalc')
plt.ylabel('Number of days of absences')
plt.show()
```