In [3]:

```python
import pandas as pd
```

That line of code is to import Pandas for me to be able to use his functions to explore, manipulate, analyze and clean the dataset.

In [4]:

```python
df = pd.read_csv('Student_Performance.csv')
```

Pandas function read_csv() is to load the dataset in Jupyter Notebook.

In [5]:

```python
duplicates = df.duplicated()
print(duplicates)
```

```
0       False
1       False
2       False
3       False
4       False
        ...
9995    False
9996    False
9997    False
9998    False
9999    False
Length: 10000, dtype: bool
```

As we can see, there are no duplicates in the Dataset.

In [6]:

```python
miss_val = df.isnull().any(axis=1)
print(miss_val)
```

```
0       False
1       False
2       False
3       False
4       False
        ...
9995    False
9996    False
9997    False
9998    False
9999    False
Length: 10000, dtype: bool
```

In [11]:

```python
df['Extracurricular Activities'] = df['Extracurricular Activities'].replace({'Yes':1, 'No':0})
```

As we can see, there is no row with a field column missing value.

In [12]:

```python
import numpy as np
import sklearn
from sklearn.linear_model import LinearRegression
```

```
#MULTIPLE LINEAR REGRESSION
```

I'm importing numpy library because, in the implementation of the multiple linear regression, I'll need it to perform mathematical operations such as vector operations and matrix manipulations. Moreover, the dependent variable and independent variables are represented as NumPy arrays in scikit-learn and Pandas, so we need to use NumPy to operate on these arrays within our model.

Now I'll be selecting the independent variables also called features and the dependent variable(target varaiable).

In [13]:

```python
X = df[['Hours Studied','Previous Scores','Extracurricular Activities','Sleep Hours']]
Y = df['Performance Index']
```

In [ ]:

```python

```

Create and fit the linear regression model

In [14]:

```python
model = LinearRegression()
model.fit(X, Y)
```

Out[14]:

```
LinearRegression()
```

Make predictions

In [15]:

```python
predictions = model.predict(X)
```

Print model summary (coefficients, R-squared, etc.)

In [17]:

```python
print(model.coef_)  # Coefficients
print(model.intercept_)  # Intercept
print(model.score(X, Y))  # R-squared value
```

```
[2.85673907 1.01868958 0.62741988 0.48194429]
-33.24005596528127
0.9879162180086278
```

In [ ]:

```
The coefficients* represent the slope/tilt of the linear relationship between the dependent variable and each one of the
independent variables.
The intercerpt represent the average expected value for the dependent variable when all independent variables are equal to Zero.
The R-values associated with each coefficient indicate whether the relationship between that predictor variable and the response is statistically significant.
A low p-value (typically < 0.05) suggests a significant relationship, while a high p-value suggests no significant effect.
```

In [ ]:

```python
                                                    #SIMPLE LINEAR REGRESSION
```

Create the model

In [19]:

```python
model = LinearRegression()
```

Fit the model (X: explanatory variable; y: response variable)

In [20]:

```python
X = df[['Hours Studied']]
y = df['Sleep Hours']
model.fit(X, y)
```

Out[20]:

```
LinearRegression()
```

Get coefficients (intercept and slope)

In [21]:

```python
intercept = model.intercept_
slope = model.coef_[0]

print(f"Intercept (b0): {intercept:.2f}")
print(f"Slope (b1): {slope:.2f}")
```

```
Intercept (b0): 6.53
Slope (b1): 0.00
```

In [ ]:

```python

```