In [1]:

```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-pyt
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all fil

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preser
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside
```

```
/kaggle/input/salary-dataset-simple-linear-regression/Salary_dataset.csv
```

In [2]:

```
class RawLR:

    def __init__(self):
        # Initialize the slope and intercept to None
        self.m = None
        self.b = None

    def fit(self, X_train, y_train):
        # Calculate the slope and intercept using the least squares method
        num = 0  # numerator of slope formula
        den = 0  # denominator of slope formula
        for i in range(X_train.shape[0]):
            # Calculate numerator and denominator of slope formula
            num += (X_train[i] - X_train.mean()) * (y_train[i] - y_train.mean())
            den += (X_train[i] - X_train.mean()) ** 2
        self.m = num / den  # Calculate slope
        self.b = y_train.mean() - self.m * X_train.mean()  # Calculate intercept
        # Print the slope and intercept for debugging purposes
        print("Slope:", self.m)
        print("Intercept:", self.b)

    def predict(self, X_test):
        # Make sure the model has been trained
        if self.m is None or self.b is None:
            raise ValueError("Model has not been trained yet")
        # Predict target values using the slope and intercept
        y_pred = self.m * X_test + self.b
        return y_pred
```

In [4]:

```
# Load the salary dataset into a pandas dataframe
df = pd.read_csv('/kaggle/input/salary-dataset-simple-linear-regression/Salary_dataset.

# Display the first few rows of the dataframe
df.head()
```

Out[4]:

	Unnamed: 0	YearsExperience	Salary
0	0	1.2	39344.0
1	1	1.4	46206.0
2	2	1.6	37732.0
3	3	2.1	43526.0
4	4	2.3	39892.0

In [5]:
```
# Extract the independent variable (YearsExperience) and the dependent variable (Salary
X = df.iloc[:, 1].values  # Years of experience
y = df.iloc[:, 2].values  # Salary
```

In [6]:
```
from sklearn.model_selection import train_test_split

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=2
```

```
/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146: UserWarning: A NumPy vers
ion >=1.16.5 and <1.23.0 is required for this version of SciPy (detected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [7]:
```
# Check the shape of the training set for the dependent variable
y_train.shape
```

Out[7]:
```
(24,)
```

In [8]:
```
# Create an instance of the RawLR class
lr = RawLR()
```

In [9]:
```
# Fit the linear regression model to the training data
lr.fit(X_train, y_train)
```

```
Slope: 9569.586885432866
Intercept: 23437.21046340505
```

In [10]:
```
# Predict the output for the first test sample using the trained model
lr.predict(X_test[0])
```

Out[10]:
```
36834.632103011056
```