OUTPUT

```
#To-Do 1: Data Understanding, Analysis & Preparation
```

Top 5 rows:

	Math	Reading	Writing
0	48	68	63
1	62	81	72
2	79	80	78
3	76	83	79
4	59	64	62

Bottom 5 rows:

	Math	Reading	Writing
995	72	74	70
996	73	86	90
997	89	87	94
998	83	82	78
999	66	66	72

```
Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   Math     1000 non-null   int64
 1   Reading  1000 non-null   int64
 2   Writing  1000 non-null   int64
dtypes: int64(3)
memory usage: 23.6 KB
```

```
Descriptive Statistics:
```

	Math	Reading	Writing
count	1000.000000	1000.000000	1000.000000
mean	67.290000	69.872000	68.616000
std	15.085008	14.657027	15.241287
min	13.000000	19.000000	14.000000
25%	58.000000	60.750000	58.000000
50%	68.000000	70.000000	69.500000
75%	78.000000	81.000000	79.000000
max	100.000000	100.000000	100.000000

```
#To-Do 2: Design Matrix without Bias
```

```
••• X shape: (1000, 2)
    Y shape: (1000,)
```

```
#To-Do 3: Train-Test Split
```

```
Training samples: 800
Testing samples: 200
```

```
#To-Do 5: Cost Function Test Case
```

```
Cost: 0.0
```

```
#To-Do 7: Gradient Descent Testing
```

```
••• Final Weights: [0.20551667 0.54295081 0.10388027]
    Final Cost: 0.05435492255484332
```

```
#To-Do 10: Main Function (Full Workflow)
```

```
••• Final Weights: [0.34811659 0.64614558]
    RMSE: 5.279823976418863 5
    R²: 0.8886354462786421
```

```
#To-Do 11: Findings (Elaborated Answer)
```

The good fit of the linear regression model may be acceptable. The model isn't overfitting since it actually does rather well on the test data and our error metrics such as RMSE don't blow up (meaning they stay more or less stable instead of going up). Yet, the model is not underfitting because it can learn some useful information about the input features (Math and Reading) that we have from the target variable (Writing). This means that the model does a good job on data it has not seen.

Significant differences in the model performances could still be seen by trying out various learning rater values. A very small learning rate results in the model converging to a global minimum very slowly; it will take many iterations for the cost function to come down. On the other hand, a very large learning rate may cause the cost function to oscillate or even increase, which in turn will cause unstable training or divergence. The best overall performance is always obtained by choosing the learning rate that leads to regular convergence with smooth reduction in error.