Excercise 1

```
import random

random.seed(0)
salaries = [round(random.random()*1000000, -3) for _ in range(100)]

df = pd.DataFrame(salaries)
df
```

	0
0	844000.0
1	758000.0
2	421000.0
3	259000.0
4	511000.0
...	...
95	917000.0
96	793000.0
97	82000.0
98	613000.0
99	486000.0

100 rows × 1 columns

Next steps: View recommended plots New interactive sheet

```
#code segment for getting the mean
mean = df.mean()
mean
```

	0
0	585690.0

dtype: float64

```
#code segment for getting the median
median = df.median()
median
```

	0
0	589000.0

dtype: float64

```
#code segment for getting the mode
mode = df.mode()
mode
```

	0
0	477000.0

```
#code segment for getting the sample variance
sample_variance = df.var()
sample_variance
```

	0
0	7.066405e+10

dtype: float64

```
#code segment for getting the standard deviation
sample_standard_deviation = df.std()
sample_standard_deviation
```

	0
0	265827.113825

dtype: float64

Excercise 2

```
#code segment for getting the range
range = df.max() - df.min()
range
```

	0
0	995000.0

dtype: float64

```
#code segment for getting the coefficient of variation
cv = (df.std() / df.mean()) * 100
cv
```

	0
0	45.386999

dtype: float64

```
#code segment for getting the Interquartile Range
q1 = df.quantile(0.25)
q3 = df.quantile(0.75)
iqr = q3 - q1
iqr
```

	0
0	413250.0

dtype: float64

```
#code segment for getting the Quartile Coefficient of Dispersion
qcd = (q3 - q1) / (q3 + q1)
qcd
```

	0
0	0.33866

dtype: float64

Excercise 3

```
import pandas as pd

diabetes = pd.read_csv('diabetes.csv')
diabetes
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPe
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

768 rows × 9 columns

Next steps: View recommended plots New interactive sheet

```
# Identify columns
diabetes.columns
```

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

```
# Identify the data types of the data
diabetes.dtypes
```

	0
Pregnancies	int64
Glucose	int64
BloodPressure	int64
SkinThickness	int64
Insulin	int64
BMI	float64
DiabetesPedigreeFunction	float64
Age	int64
Outcome	int64

dtype: object

```
# Display the total number of records
diabetes.shape[0]
```

```
768
```

```
# Display the first 20 records
diabetes.head(20)
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPed
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
5	5	116	74	0	0	25.6	
6	3	78	50	32	88	31.0	
7	10	115	0	0	0	35.3	
8	2	197	70	45	543	30.5	
9	8	125	96	0	0	0.0	
10	4	110	92	0	0	37.6	
11	10	168	74	0	0	38.0	
12	10	139	80	0	0	27.1	
13	1	189	60	23	846	30.1	
14	5	166	72	19	175	25.8	
15	7	100	0	0	0	30.0	
16	0	118	84	47	230	45.8	
17	7	107	74	0	0	29.6	
18	1	103	30	38	83	43.3	
19	1	115	70	30	96	34.6	

Next steps: View recommended plots New interactive sheet

```
# Display the last 20 records
diabetes.tail(20)
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPe
748	3	187	70	22	200	36.4	
749	6	162	62	0	0	24.3	
750	4	136	70	0	0	31.2	
751	1	121	78	39	74	39.0	
752	3	108	62	24	0	26.0	
753	0	181	88	44	510	43.3	
754	8	154	78	32	0	32.4	
755	1	128	88	39	110	36.5	
756	7	137	90	41	0	32.0	
757	0	123	72	0	0	36.3	
758	1	106	76	0	0	37.5	
759	6	190	92	0	0	35.5	
760	2	88	58	26	16	28.4	
761	9	170	74	31	0	44.0	
762	9	89	62	0	0	22.5	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

```
# Change the Outcome column to Diagnosis
dcopy = diabetes.rename(columns = {'Outcome':'Diagnosis'})
dcopy
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPe
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

768 rows × 9 columns

Next steps: View recommended plots New interactive sheet

```
# Create a new column Classification that display "Diabetes" if the value of outcome is 1 , otherwise "No Diabetes"
dcopy['Classification'] = (dcopy['Diagnosis'] == 1).map({True: 'Diabetes', False: 'No Diabetes'})
dcopy
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPe
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

768 rows × 10 columns

Next steps: View recommended plots New interactive sheet

```
# Create a new dataframe "withDiabetes" that gathers data with diabetes
newdata = dcopy[dcopy['Diagnosis']==1]
withDiabetes = pd.DataFrame(newdata)
withDiabetes
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPe
0	6	148	72	35	0	33.6	
2	8	183	64	0	0	23.3	
4	0	137	40	35	168	43.1	
6	3	78	50	32	88	31.0	
8	2	197	70	45	543	30.5	
...	...	...	...	...	...	...	
755	1	128	88	39	110	36.5	
757	0	123	72	0	0	36.3	
759	6	190	92	0	0	35.5	
761	9	170	74	31	0	44.0	
766	1	126	60	0	0	30.1	

268 rows × 10 columns

Next steps: View recommended plots New interactive sheet

```
# Create a new dataframe "noDiabetes" thats gathers data with no diabetes
newdata2 = dcopy[dcopy['Diagnosis']==0]
noDiabetes = pd.DataFrame(newdata2)
noDiabetes
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPe
1	1	85	66	29	0	26.6	
3	1	89	66	23	94	28.1	
5	5	116	74	0	0	25.6	
7	10	115	0	0	0	35.3	
10	4	110	92	0	0	37.6	
...	...	...	...	...	...	...	
762	9	89	62	0	0	22.5	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
767	1	93	70	31	0	30.4	

500 rows × 10 columns

Next steps: View recommended plots New interactive sheet

```
# Create a new dataframe "Pedia" that gathers data with age 0 to 19
newdata3 = dcopy[dcopy['Age'] <= 19]
Pedia = pd.DataFrame(newdata3)
Pedia
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigr

```
# Create a new dataframe "Adult" that gathers data with age greater than 19
newdata4 = dcopy[dcopy['Age'] >= 19]
Adult = pd.DataFrame(newdata4)
Adult
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPe
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	

Next steps: View recommended plots New interactive sheet

```
import numpy as np

# Use numpy to get the average age and glucose value.

avg_age = np.mean(dcopy['Age'])
avg_glucose = np.mean(dcopy['Glucose'])

print("Average Age:", avg_age)
print("Average Glucose:", avg_glucose)
```

```
Average Age: 33.240885416666664
Average Glucose: 120.89453125
```

```
# Use numpy to get the median age and glucose value.

median_age = np.median(dcopy['Age'])
median_glucose = np.median(dcopy['Glucose'])

print("Median Age:", median_age)
print("Median Glucose:", median_glucose)
```

```
Median Age: 29.0
Median Glucose: 117.0
```

```
# Use numpy to get the middle values of glucose and age.

middle_age = np.median(dcopy['Age'])
middle_glucose = np.median(dcopy['Glucose'])

print("Middle Age:", middle_age)
print("Middle Glucose:", middle_glucose)
```

```
Middle Age: 29.0
Middle Glucose: 117.0
```