In [2]:

```
import pandas as pd
import matplotlib
import seaborn as sns
import matplotlib.pyplot as plt
```

In [3]:

```
# # |Load data
dataw = pd.read_csv("G:\\CS NOTES\\2.2\\SCIENTIFIC COMPUTING\\Assignment\\pandas_into\\Diabetes Data.csv")
dataw
```

Out[3]:

	Pregnant	Glucose	Diastolic_BP	Skin_Fold	Serum_Insulin	BMI	Diabetes_Pedigree	Age	Class
0	6	148.0	72.0	35.0	NaN	33.6	0.627	50	1
1	1	85.0	66.0	29.0	NaN	26.6	0.351	31	0
2	8	183.0	64.0	NaN	NaN	23.3	0.672	32	1
3	1	89.0	66.0	23.0	94.0	28.1	0.167	21	0
4	0	137.0	40.0	35.0	168.0	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101.0	76.0	48.0	180.0	32.9	0.171	63	0
764	2	122.0	70.0	27.0	NaN	36.8	0.340	27	0
765	5	121.0	72.0	23.0	112.0	26.2	0.245	30	0
766	1	126.0	60.0	NaN	NaN	30.1	0.349	47	1
767	1	93.0	70.0	31.0	NaN	30.4	0.315	23	0

768 rows × 9 columns

In [4]:

```
# getting the dataset information
dataw.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Pregnant           768 non-null    int64  
 1   Glucose            763 non-null    float64
 2   Diastolic_BP       733 non-null    float64
 3   Skin_Fold          541 non-null    float64
 4   Serum_Insulin      394 non-null    float64
 5   BMI                757 non-null    float64
 6   Diabetes_Pedigree  768 non-null    float64
 7   Age                768 non-null    int64  
 8   Class              768 non-null    int64  
dtypes: float64(6), int64(3)
memory usage: 54.1 KB
```

In [5]:

```
# checking the first 5 rows of the dataset
dataw.head()
```

Out[5]:

	Pregnant	Glucose	Diastolic_BP	Skin_Fold	Serum_Insulin	BMI	Diabetes_Pedigree	Age	Class
0	6	148.0	72.0	35.0	NaN	33.6	0.627	50	1
1	1	85.0	66.0	29.0	NaN	26.6	0.351	31	0
2	8	183.0	64.0	NaN	NaN	23.3	0.672	32	1
3	1	89.0	66.0	23.0	94.0	28.1	0.167	21	0
4	0	137.0	40.0	35.0	168.0	43.1	2.288	33	1

In [6]:

```
# |checking the last 5 rows of the dataset
dataw.tail()
```

Out[6]:

	Pregnant	Glucose	Diastolic_BP	Skin_Fold	Serum_Insulin	BMI	Diabetes_Pedigree	Age	Class
763	10	101.0	76.0	48.0	180.0	32.9	0.171	63	0
764	2	122.0	70.0	27.0	NaN	36.8	0.340	27	0
765	5	121.0	72.0	23.0	112.0	26.2	0.245	30	0
766	1	126.0	60.0	NaN	NaN	30.1	0.349	47	1
767	1	93.0	70.0	31.0	NaN	30.4	0.315	23	0

In [7]:

```
# checking the size of the data
dataw.shape
```

Out[7]:

```
(768, 9)
```

In [8]:
```
# checking the data types for each column of ther dataset
dataw.dtypes
```

Out[8]:
```
Pregnant               int64
Glucose              float64
Diastolic_BP         float64
Skin_Fold            float64
Serum_Insulin        float64
BMI                  float64
Diabetes_Pedigree    float64
Age                    int64
Class                  int64
dtype: object
```

In [9]:
```
# dataw["Pregnant"].value_counts()
```

In [10]:
```
# getting summary Statistics for each columns
dataw.describe(include='all')
```

Out[10]:

	Pregnant	Glucose	Diastolic_BP	Skin_Fold	Serum_Insulin	BMI	Diabetes_Pedigree	Age	Cla
count	768.000000	763.000000	733.000000	541.000000	394.000000	757.000000	768.000000	768.000000	768.0000
mean	3.845052	121.686763	72.405184	29.153420	155.548223	32.457464	0.471876	33.240885	0.3489
std	3.369578	30.535641	12.382158	10.476982	118.775855	6.924988	0.331329	11.760232	0.4769
min	0.000000	44.000000	24.000000	7.000000	14.000000	18.200000	0.078000	21.000000	0.0000
25%	1.000000	99.000000	64.000000	22.000000	76.250000	27.500000	0.243750	24.000000	0.0000
50%	3.000000	117.000000	72.000000	29.000000	125.000000	32.300000	0.372500	29.000000	0.0000
75%	6.000000	141.000000	80.000000	36.000000	190.000000	36.600000	0.626250	41.000000	1.0000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.0000

In [11]:
```
# checking count of missing values per column
dataw.isnull().sum()
```

Out[11]:
```
Pregnant               0
Glucose                5
Diastolic_BP          35
Skin_Fold            227
Serum_Insulin        374
BMI                   11
Diabetes_Pedigree      0
Age                    0
Class                  0
dtype: int64
```

In [12]:
```
# filling the missing values in the dataset with the mean
dataw.fillna(dataw.mean(numeric_only=True), inplace=True)
dataw
# dataw.fillna(dataw.mean(), inplace=True)
# dataw
```

Out[12]:

	Pregnant	Glucose	Diastolic_BP	Skin_Fold	Serum_Insulin	BMI	Diabetes_Pedigree	Age	Class
0	6	148.0	72.0	35.00000	155.548223	33.6	0.627	50	1
1	1	85.0	66.0	29.00000	155.548223	26.6	0.351	31	0
2	8	183.0	64.0	29.15342	155.548223	23.3	0.672	32	1
3	1	89.0	66.0	23.00000	94.000000	28.1	0.167	21	0
4	0	137.0	40.0	35.00000	168.000000	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101.0	76.0	48.00000	180.000000	32.9	0.171	63	0
764	2	122.0	70.0	27.00000	155.548223	36.8	0.340	27	0
765	5	121.0	72.0	23.00000	112.000000	26.2	0.245	30	0
766	1	126.0	60.0	29.15342	155.548223	30.1	0.349	47	1
767	1	93.0	70.0	31.00000	155.548223	30.4	0.315	23	0

768 rows × 9 columns

In [13]:
```
# calculation of key statistics
dataw.mean()
```

Out[13]:

```
Pregnant              3.845052
Glucose             121.686763
Diastolic_BP         72.405184
Skin_Fold            29.153420
Serum_Insulin       155.548223
BMI                  32.457464
Diabetes_Pedigree     0.471876
Age                  33.240885
Class                 0.348958
dtype: float64
```

In [14]:

```
dataw.median()
```

Out[14]:

```
Pregnant              3.000000
Glucose             117.000000
Diastolic_BP         72.202592
Skin_Fold            29.153420
Serum_Insulin       155.548223
BMI                  32.400000
Diabetes_Pedigree     0.372500
Age                  29.000000
Class                 0.000000
dtype: float64
```

In [15]:

```
dataw.mode()
# dataw['Glucose'].mode()
```

Out[15]:

	Pregnant	Glucose	Diastolic_BP	Skin_Fold	Serum_Insulin	BMI	Diabetes_Pedigree	Age	Class
0	1.0	99.0	70.0	29.15342	155.548223	32.0	0.254	22.0	0.0
1	NaN	100.0	NaN	NaN	NaN	NaN	0.258	NaN	NaN

In [16]:

```
dataw.std()
```

Out[16]:

```
Pregnant              3.369578
Glucose              30.435949
Diastolic_BP         12.096346
Skin_Fold             8.790942
Serum_Insulin        85.021108
BMI                   6.875151
Diabetes_Pedigree     0.331329
Age                  11.760232
Class                 0.476951
dtype: float64
```

In [17]:

```
# detecting outliers/checking if the dataset has outliers
Q1=dataw.quantile(0.25)
Q3=dataw.quantile(0.75)
IQR=Q3-Q1

upper_limit=Q3+1.5*IQR
lower_limit=Q1-1.5*IQR
outliers = (dataw < lower_limit) | (dataw > upper_limit)
outliers
```

Out[17]:

	Pregnant	Glucose	Diastolic_BP	Skin_Fold	Serum_Insulin	BMI	Diabetes_Pedigree	Age	Class
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	True	False	False
...	...	...	...	...	...	...	...	...	...
763	False	False	False	True	False	False	False	False	False
764	False	False	False	False	False	False	False	False	False
765	False	False	False	False	False	False	False	False	False
766	False	False	False	False	False	False	False	False	False
767	False	False	False	False	False	False	False	False	False

768 rows × 9 columns

In [25]:

```
# Removing outliers for column and replace them with column's mean
data_outliers_free = dataw.mask((dataw < lower_limit) | (dataw > upper_limit), dataw.mean(axis=0), axis=1)
data_outliers_free
```

Out[25]:

	Pregnant	Glucose	Diastolic_BP	Skin_Fold	Serum_Insulin	BMI	Diabetes_Pedigree	Age	Class
0	6.0	148.0	72.0	35.00000	155.548223	33.6	0.627000	50.0	1
1	1.0	85.0	66.0	29.00000	155.548223	26.6	0.351000	31.0	0
2	8.0	183.0	64.0	29.15342	155.548223	23.3	0.672000	32.0	1
3	1.0	89.0	66.0	23.00000	94.000000	28.1	0.167000	21.0	0
4	0.0	137.0	40.0	35.00000	168.000000	43.1	0.471876	33.0	1
...	...	...	...	...	...	...	...	...	...
763	10.0	101.0	76.0	29.15342	180.000000	32.9	0.171000	63.0	0
764	2.0	122.0	70.0	27.00000	155.548223	36.8	0.340000	27.0	0
765	5.0	121.0	72.0	23.00000	112.000000	26.2	0.245000	30.0	0
766	1.0	126.0	60.0	29.15342	155.548223	30.1	0.349000	47.0	1
767	1.0	93.0	70.0	31.00000	155.548223	30.4	0.315000	23.0	0

768 rows × 9 columns

In [ ]:

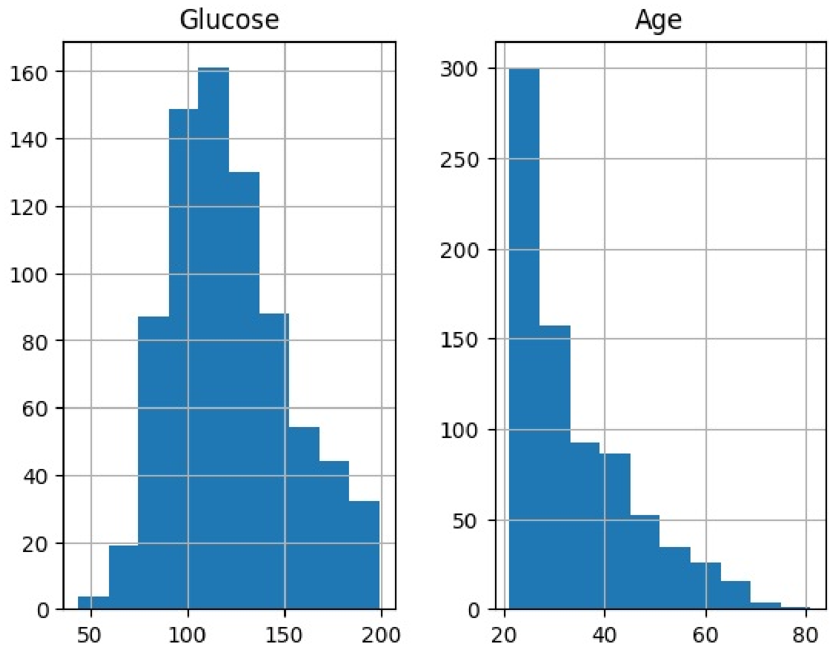
```
# Dataset's Histogram representation for every column separately
# dataw.hist()
```

In [ ]:

```
# Selecting multiple columns of the dataset and visualizing their data using a histogram
dataw[['Glucose', "Age",]].hist()
```

Out[ ]:

```
array([[<Axes: title={'center': 'Glucose'}>,
        <Axes: title={'center': 'Age'}>]], dtype=object)
```

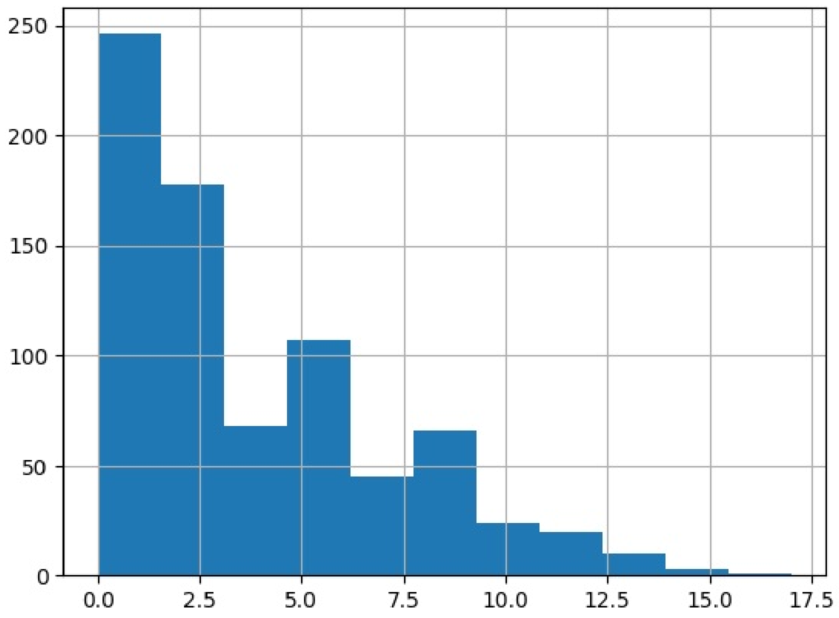


In [ ]:

```
# Selecting a single column (Pregnant column) and visualise it's data
dataw['Pregnant'].hist(bins=11)
```

Out[ ]:

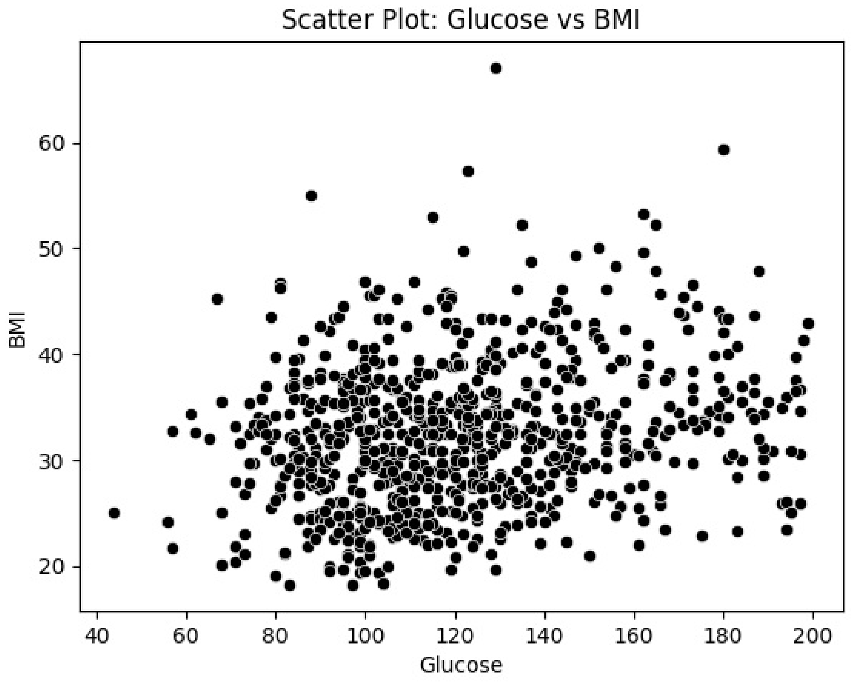
```
<Axes: >
```



In [ ]:

```
# scatterplot1
sns.scatterplot(data=dataw, x="Glucose", y="BMI", color="black")

plt.title("Scatter Plot: Glucose vs BMI")
plt.show()
```



In [ ]:

```
# # scatterplot2
# sns.scatterplot(data=dataw, x="Glucose", y="Pregnant", color="grey")
# plt.title("Scatter Plot: Glucose vs Pregnant")
# plt.show()
```

Out[ ]:

```
Pregnant               1.00000
Glucose               99.75000
Diastolic_BP          64.00000
Skin_Fold             25.00000
Serum_Insulin        121.50000
BMI                   27.50000
Diabetes_Pedigree      0.24375
Age                   24.00000
Class                  0.00000
Name: 0.25, dtype: float64
```

In [ ]:

```
# Compute the correlation matrix
corr_matrix = dataw.corr()

corr_matrix
```

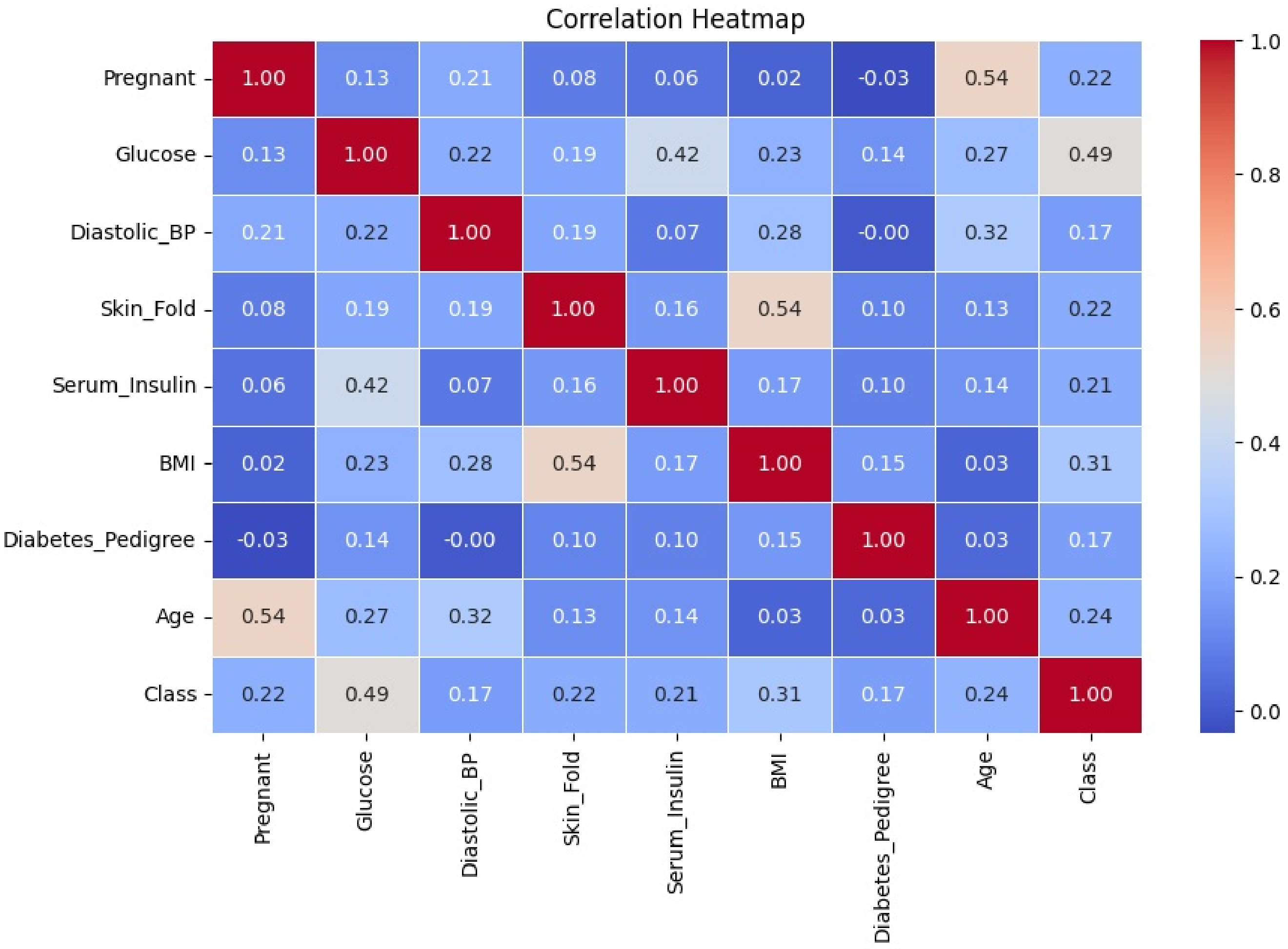
Out[ ]:

	Pregnant	Glucose	Diastolic_BP	Skin_Fold	Serum_Insulin	BMI	Diabetes_Pedigree	Age	Cl
Pregnant	1.000000	0.127911	0.208522	0.082989	0.056027	0.021565	-0.033523	0.544341	0.221
Glucose	0.127911	1.000000	0.218367	0.192991	0.420157	0.230941	0.137060	0.266534	0.492
Diastolic_BP	0.208522	0.218367	1.000000	0.192816	0.072517	0.281268	-0.002763	0.324595	0.166
Skin_Fold	0.082989	0.192991	0.192816	1.000000	0.158139	0.542398	0.100966	0.127872	0.215
Serum_Insulin	0.056027	0.420157	0.072517	0.158139	1.000000	0.166586	0.098634	0.136734	0.214
BMI	0.021565	0.230941	0.281268	0.542398	0.166586	1.000000	0.153400	0.025519	0.311
Diabetes_Pedigree	-0.033523	0.137060	-0.002763	0.100966	0.098634	0.153400	1.000000	0.033561	0.173
Age	0.544341	0.266534	0.324595	0.127872	0.136734	0.025519	0.033561	1.000000	0.238
Class	0.221898	0.492928	0.166074	0.215299	0.214411	0.311924	0.173844	0.238356	1.000

In [ ]:

```
# correlation heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(corr_matrix, annot=True, cmap="coolwarm", fmt=".2f", linewidths=0.5)

plt.title("Correlation Heatmap")
plt.show()
```


Correlation Heatmap
Pregnant 1.00 0.13 0.21 0.08 0.06 0.02 -0.03 0.54 0.22
Glucose 0.13 1.00 0.22 0.19 0.42 0.23 0.14 0.27 0.49
Diastolic_BP 0.21 0.22 1.00 0.19 0.07 0.28 -0.00 0.32 0.17
Skin_Fold 0.08 0.19 0.19 1.00 0.16 0.54 0.10 0.13 0.22
Serum_Insulin 0.06 0.42 0.07 0.16 1.00 0.17 0.10 0.14 0.21
BMI 0.02 0.23 0.28 0.54 0.17 1.00 0.15 0.03 0.31
Diabetes_Pedigree -0.03 0.14 -0.00 0.10 0.10 0.15 1.00 0.03 0.17
Age 0.54 0.27 0.32 0.13 0.14 0.03 0.03 1.00 0.24
Class 0.22 0.49 0.17 0.22 0.21 0.31 0.17 0.24 1.00
Pregnant
Glucose
Diastolic_BP
Skin_Fold
Serum_Insulin
BMI
Diabetes_Pedigree
Age
Class