MeriSKILL

Project 2 : Diabetes patients Analysis

the Attribute information:

1.Pregnancies: Number of times pregnant

2.Glucose: Plasma glucose concentration a 2 hours in an oral glucose tolerance test

3.Blood pressure: Diastolic blood pressure (mm Hg)

4.SkinThickness: Triceps skinfold thickness (mm)

5.Insulin: 2-Hour serum insulin (mu U/ml) test

6.BMI: Body mass index (weight in kg/(height in m)^2)

7.DiabetesPedigreeFunction: A function that scores the likelihood of diabetes based on family history

8.Age: Age in years

9.Outcome: Class variable (0: the person is not diabetic or 1: the person is diabetic)

Reading the data

In [1]:
```
# import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [2]:
```
# load the Dataset
df = pd.read_csv(r"D:\intership\diabetes.csv")
```

In [3]:
```
df.shape
```

Out[3]:
```
(768, 9)
```

In [4]:
```
df.columns
```

Out[4]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [5]:
```
df.head(10)
```

Out[5]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
5	5	116	74	0	0	25.6	0.201	30	0
6	3	78	50	32	88	31.0	0.248	26	1
7	10	115	0	0	0	35.3	0.134	29	0
8	2	197	70	45	543	30.5	0.158	53	1
9	8	125	96	0	0	0.0	0.232	54	1

Variable Identification

In [6]:
```
df.dtypes
```

Out[6]:
```
Pregnancies                   int64
Glucose                       int64
BloodPressure                 int64
SkinThickness                 int64
Insulin                       int64
BMI                         float64
DiabetesPedigreeFunction    float64
Age                           int64
Outcome                       int64
dtype: object
```

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [9]:
```
df.nunique()
```

Out[9]:
```
Pregnancies                  17
Glucose                     136
BloodPressure                47
SkinThickness                51
Insulin                     186
BMI                         248
DiabetesPedigreeFunction    517
Age                          52
Outcome                       2
dtype: int64
```

In [10]:
```
# find the how many times the female hae been pregnent
df["Pregnancies"].unique()
```

Out[10]:
```
array([ 6,  1,  8,  0,  5,  3, 10,  2,  4,  7,  9, 11, 13, 15, 17, 12, 14],
      dtype=int64)
```

In [11]:
```
#1 patients are 17 times Pregnent
#135 patients are 1 times Pregnent
#111 patients are 0 times Pregnent
df["Pregnancies"].value_counts()
```

Out[11]:
```
1     135
0     111
2     103
3      75
4      68
5      57
6      50
7      45
8      38
9      28
10     24
11     11
13     10
12      9
14      2
15      1
17      1
Name: Pregnancies, dtype: int64
```

In [12]:
```
# to check in percentage
df["Pregnancies"].value_counts()/len(df["Pregnancies"])
```

Out[12]:
```
1     0.175781
0     0.144531
2     0.134115
3     0.097656
4     0.088542
5     0.074219
6     0.065104
7     0.058594
8     0.049479
9     0.036458
10    0.031250
11    0.014323
13    0.013021
12    0.011719
14    0.002604
15    0.001302
17    0.001302
Name: Pregnancies, dtype: float64
```

In [13]:
```
df["BloodPressure"].unique()
```

Out[13]:
```
array([ 72,  66,  64,  40,  74,  50,   0,  70,  96,  92,  80,  60,  84,
        30,  88,  90,  94,  76,  82,  75,  58,  78,  68, 110,  56,  62,
        85,  86,  48,  44,  65, 108,  55, 122,  54,  52,  98, 104,  95,
        46, 102, 100,  61,  24,  38, 106, 114], dtype=int64)
```

In [14]:
```
# 35 patient who have zero blood pressure
df["BloodPressure"].value_counts()
```

Out[14]:
```
70     57
74     52
78     45
68     45
72     44
64     43
80     40
76     39
60     37
0      35
62     34
66     30
82     30
88     25
84     23
90     22
86     21
58     21
50     13
56     12
52     11
54     11
75      8
92      8
65      7
85      6
94      6
48      5
96      4
44      4
100     3
106     3
98      3
110     3
55      2
108     2
104     2
46      2
30      2
122     1
95      1
102     1
61      1
24      1
38      1
40      1
114     1
Name: BloodPressure, dtype: int64
```

In [15]:
```
# to check in percentage
df["BloodPressure"].value_counts()/len(df["BloodPressure"])
```

Out[15]:
```
70     0.074219
74     0.067708
78     0.058594
68     0.058594
72     0.057292
64     0.055990
80     0.052083
76     0.050781
60     0.048177
0      0.045573
62     0.044271
66     0.039062
82     0.039062
88     0.032552
84     0.029948
90     0.028646
86     0.027344
58     0.027344
50     0.016927
56     0.015625
52     0.014323
54     0.014323
75     0.010417
92     0.010417
65     0.009115
85     0.007812
94     0.007812
48     0.006510
96     0.005208
44     0.005208
100    0.003906
106    0.003906
98     0.003906
110    0.003906
55     0.002604
108    0.002604
104    0.002604
46     0.002604
30     0.002604
122    0.001302
95     0.001302
102    0.001302
61     0.001302
24     0.001302
38     0.001302
40     0.001302
114    0.001302
Name: BloodPressure, dtype: float64
```

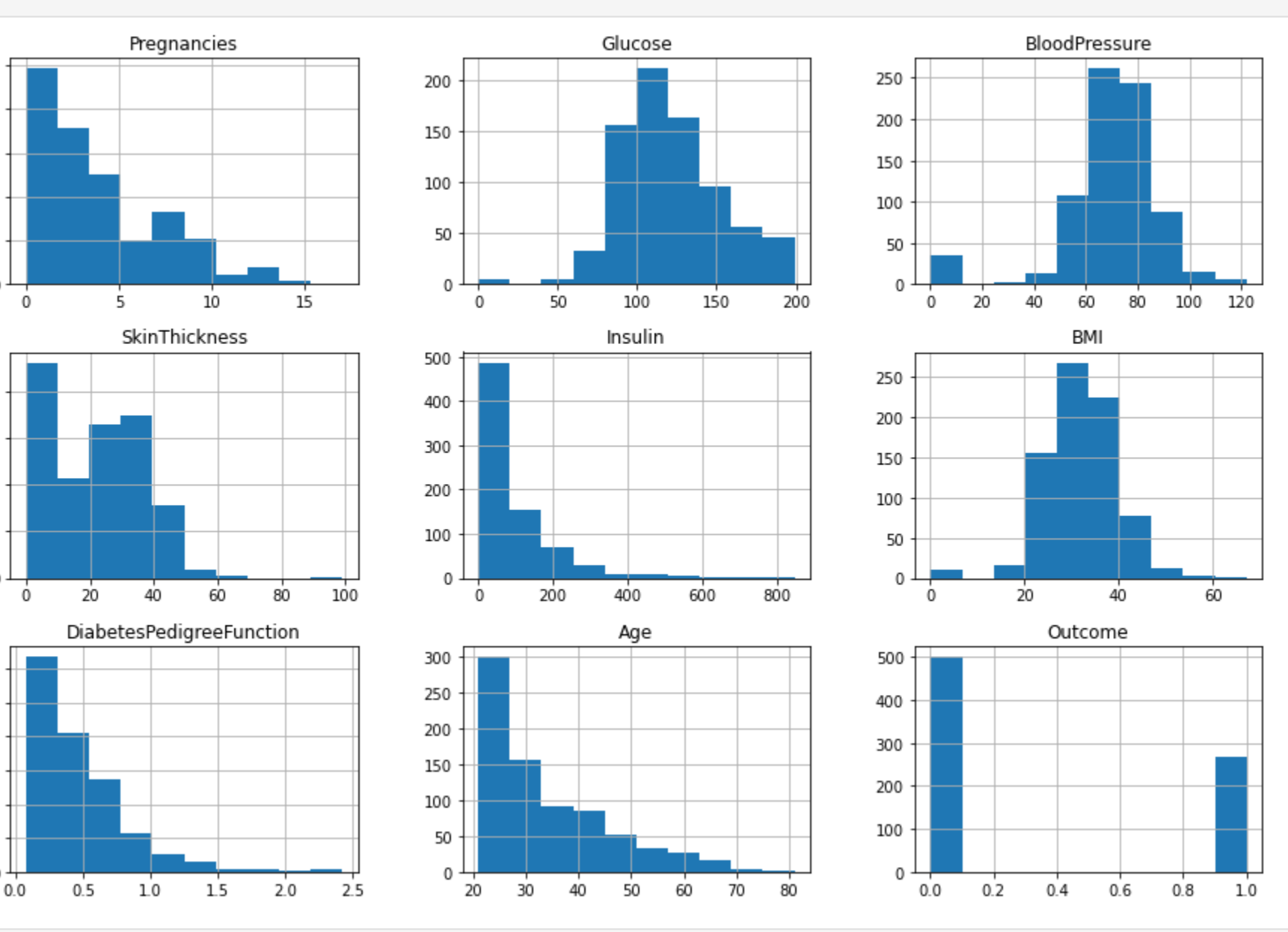
In [16]:
```
# From the table
# The pregnancy numbers appear to be normally distributed whereas the others seem to be rightly skewed.
# The mean and std deviation of pregnancies are more or less the same as opposed to the others
# Highest glucose levels is 199, pregnancies 17 and BMI 67
df.describe()
```

Out[16]:

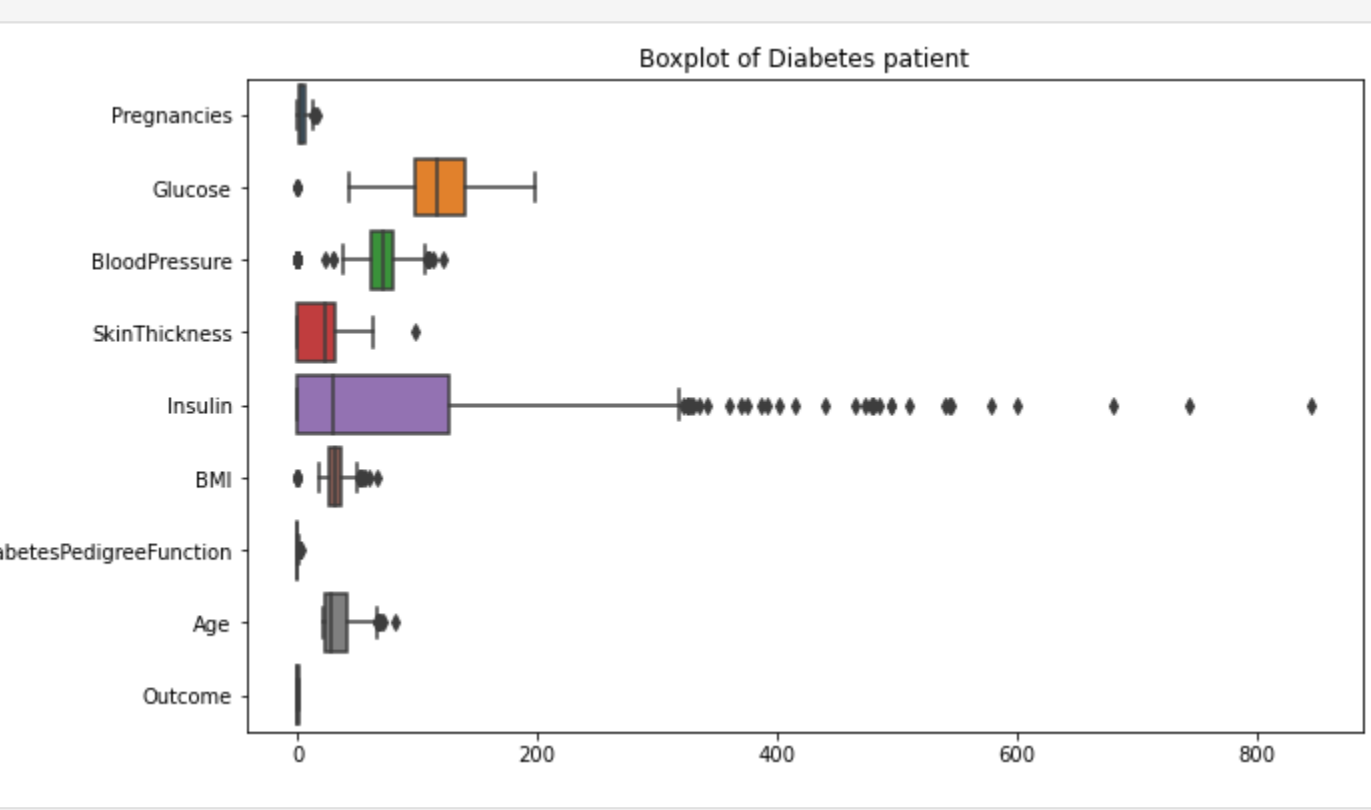
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

Data Visualization

In [17]:
```
#Histrograms of each feature
df.hist(bins=10,figsize=(15,10))
plt.legend
plt.show()
```

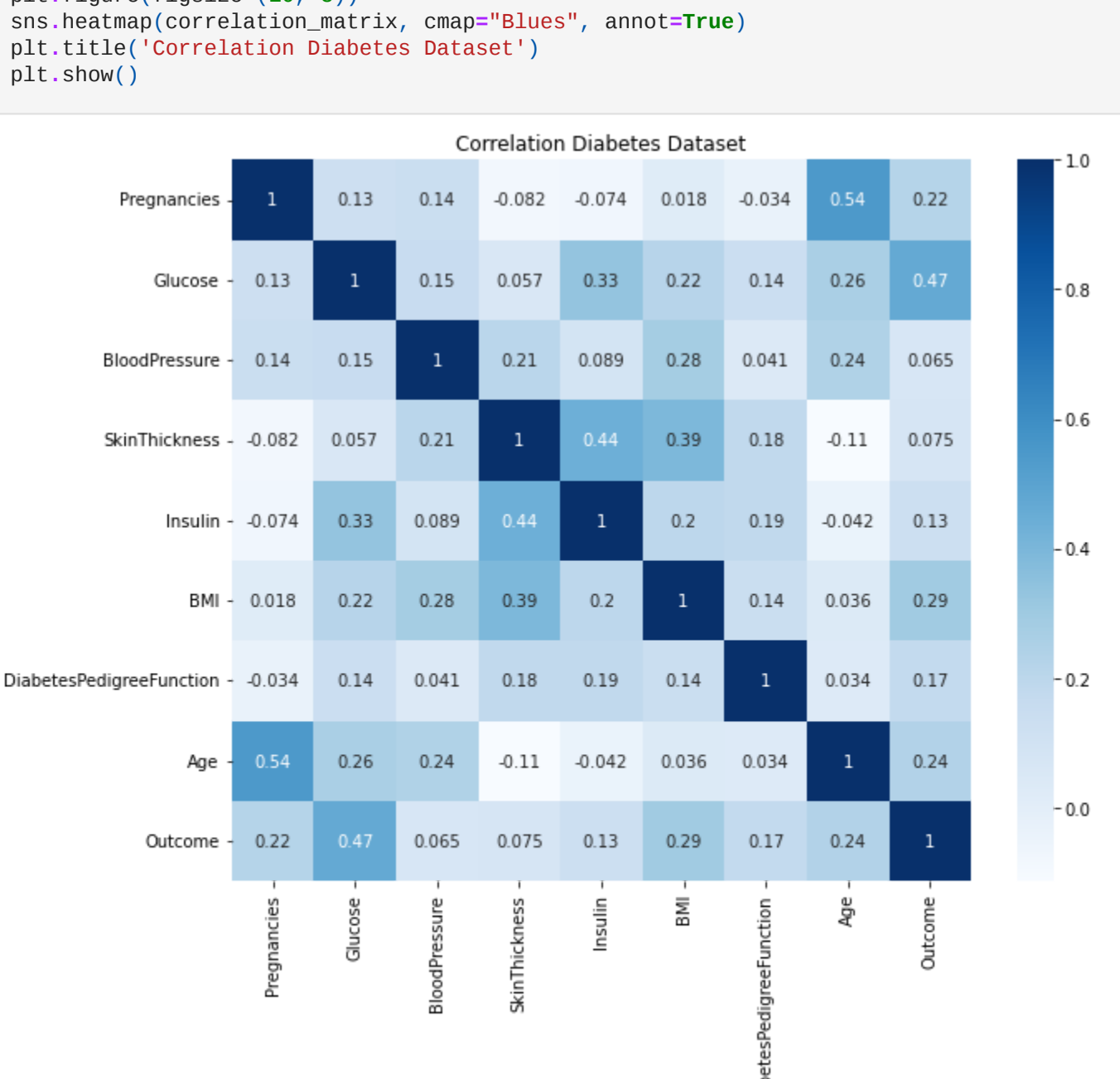


In [18]:
```
# Boxplot to detect outliers
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, orient='h')
plt.title('Boxplot of Diabetes patient')
plt.show()
```



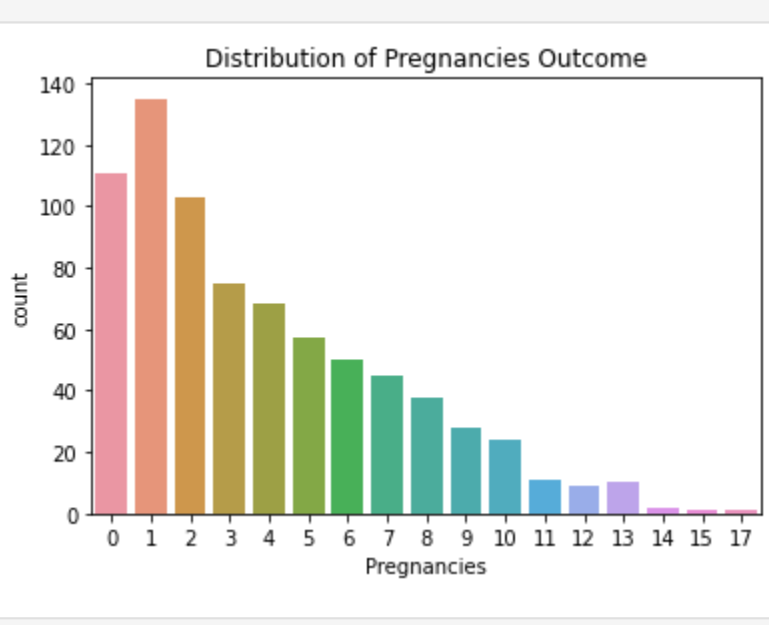
In [19]:
```
#Close to 1 indicates a very good relationship
#Close to -1 indicates a very poor relationship
#From the above graph, we can see that the relationship is very strong for the below features
#Age-Pregnancies
#SkinThickness-BMI
#Glucose-Insulin

correlation_matrix = df.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, cmap="Blues", annot=True)
plt.title('Correlation Diabetes Dataset')
plt.show()
```

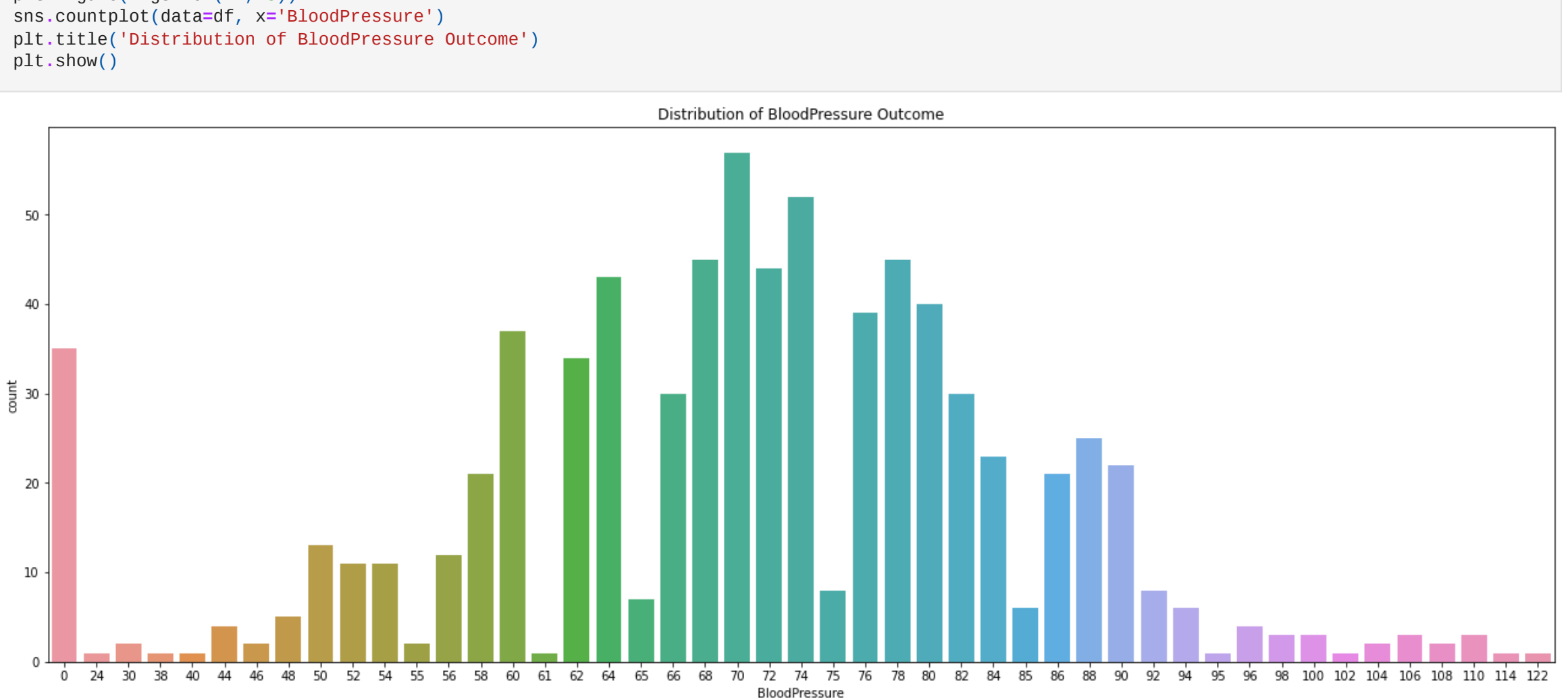


In [20]:
```
#we have observed in Pair plot analysis, we also found a strong relationship between age and pregnancy, Glucose Insulin
sns.pairplot(df, hue='Outcome')  # Pairplot for numeric variables, color by Outcome
plt.show()
```

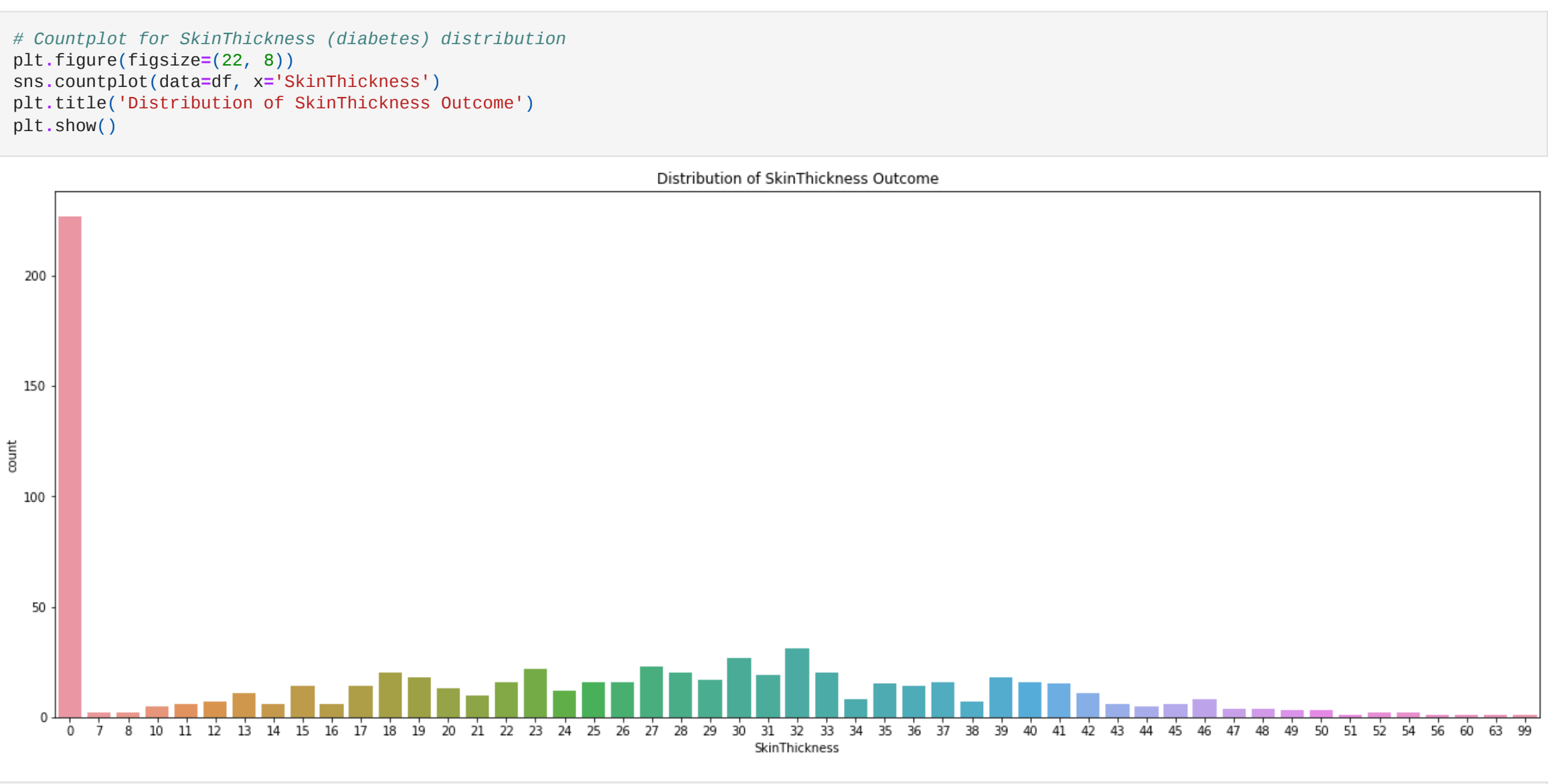
In [21]:
```
# Countplot for Pregnancies (diabetes) distribution
plt.figure(figsize=(6, 4))
sns.countplot(data=df, x='Pregnancies')
plt.title('Distribution of Pregnancies Outcome')
plt.show()
```



In [22]:
```
#Countplot for BloodPressure (diabetes) distribution
plt.figure(figsize=(22, 8))
sns.countplot(data=df, x='BloodPressure')
plt.title('Distribution of BloodPressure Outcome')
plt.show()
```

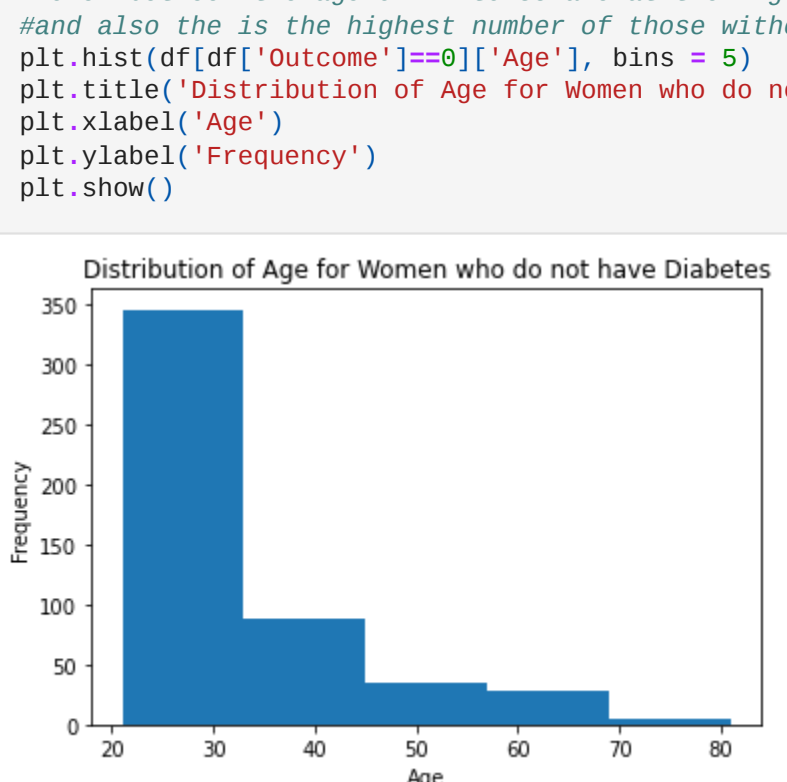


In [23]:
```
# Countplot for SkinThickness (diabetes) distribution
plt.figure(figsize=(22, 8))
sns.countplot(data=df, x='SkinThickness')
plt.title('Distribution of SkinThickness Outcome')
plt.show()
```

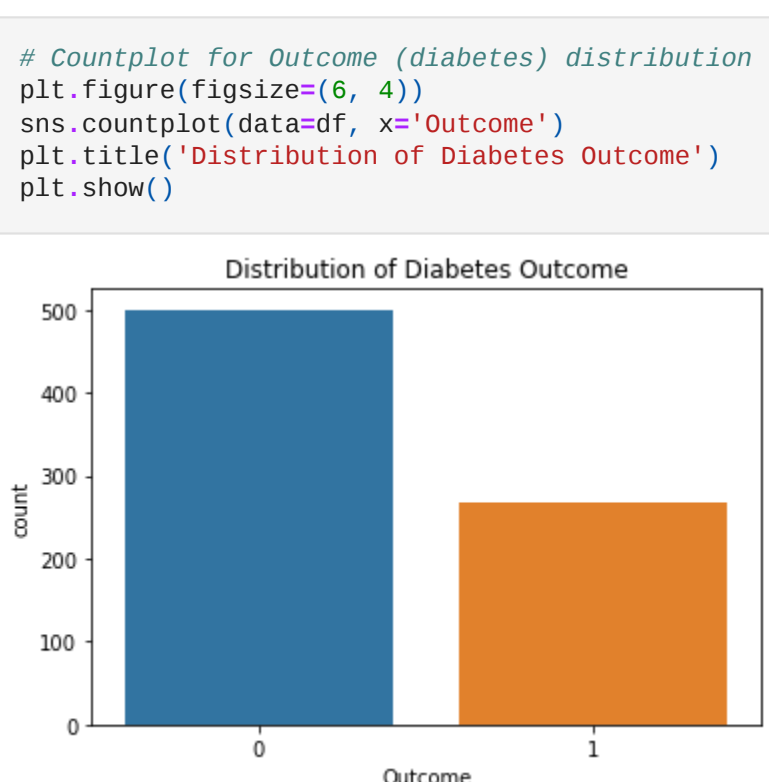


In [24]:
```
# Understanding the number of women in different age groups with diabetes.
# all the women with diabetes most are from the age between 22 to 30.
#The frequency of women with diabetes decreases as age increases.
plt.hist(df[df['Outcome']==1]['Age'], bins = 5)
plt.title('Distribution of Age for Women who has Diabetes')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```

In [25]:
```
#Understanding the number of women in different age groups without diabetes.
#The highest number of Women without diabetes range between ages 22 to 33.
#Women between the age of 22 to 35 are at the highest risk of diabetes
#and also the is the highest number of those without diabetes.
plt.hist(df[df['Outcome']==0]['Age'], bins = 5)
plt.title('Distribution of Age for Women who do not have Diabetes')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```



In [26]:
```
# Countplot for Outcome (diabetes) distribution
plt.figure(figsize=(6, 4))
sns.countplot(data=df, x='Outcome')
plt.title('Distribution of Diabetes Outcome')
plt.show()
```



We have seen that the behaviors of those with diabetes are around 35%, while those without diabetes are around 65%.