jupyter task no 6.1 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [2]:
```
import pandas as pd
# 1. Read the diabetes dataset
diabetes_data = pd.read_csv('diabetes.csv')
```

In [3]:
```
# 2. Display top 5 records
top_5_records = diabetes_data.head(5)
print("Top 5 records:")
print(top_5_records)
```

```
Top 5 records:
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
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

   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1
```

jupyter task no 6.1 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [4]:
```
# 3. Display bottom 5 records
bottom_5_records = diabetes_data.tail(5)
print("\nBottom 5 records:")
print(bottom_5_records)
```

```
Bottom 5 records:
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
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

     DiabetesPedigreeFunction  Age  Outcome
763                     0.171   63        0
764                     0.340   27        0
765                     0.245   30        0
766                     0.349   47        1
767                     0.315   23        0
```

In [5]:
```
# 4. Display statistical information of dataset
statistical_info = diabetes_data.describe()
print("\nStatistical information:")
print(statistical_info)
```

jupyter task no 6.1 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

```
Statistical information:
       Pregnancies     Glucose  BloodPressure  SkinThickness     Insulin  \
count   768.000000  768.000000     768.000000     768.000000  768.000000
mean      3.845052  120.894531      69.105469      20.536458   79.799479
std       3.369578   31.972618      19.355807      15.952218  115.244002
min       0.000000    0.000000       0.000000       0.000000    0.000000
25%       1.000000   99.000000      62.000000       0.000000    0.000000
50%       3.000000  117.000000      72.000000      23.000000   30.500000
75%       6.000000  140.250000      80.000000      32.000000  127.250000
max      17.000000  199.000000     122.000000      99.000000  846.000000

              BMI  DiabetesPedigreeFunction         Age     Outcome
count  768.000000                768.000000  768.000000  768.000000
mean    31.992578                  0.471876   33.240885    0.348958
std      7.884160                  0.331329   11.760232    0.476951
min      0.000000                  0.078000   21.000000    0.000000
25%     27.300000                  0.243750   24.000000    0.000000
50%     32.000000                  0.372500   29.000000    0.000000
75%     36.600000                  0.626250   41.000000    1.000000
max     67.100000                  2.420000   81.000000    1.000000
```

In [6]:
```
# 5. Display more information about all attributes
attribute_info = diabetes_data.info()
print("\nAttribute information:")
print(attribute_info)
```

jupyter task no 6.1 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

```
<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

Attribute information:
None
```

In [7]:
```
# 6. Display the list of columns for the given dataset
columns_list = diabetes_data.columns
print("\nList of columns:")
print(columns_list)
```

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

In [8]:
```
# 7. Display number of records and attributes
num_records, num_attributes = diabetes_data.shape
print("\nNumber of records:", num_records)
print("Number of attributes:", num_attributes)
```

```
Number of records: 768
Number of attributes: 9
```

jupyter task no 6.1 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [11]:
```
# 8. Check for null values and remove them if found
null_values = diabetes_data.isnull().sum()
print("\nNull values:")
print(null_values)
# Remove rows with null values
diabetes_data = diabetes_data.dropna()
```

```
Null values:
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [12]:
```
# 9. Check for duplicate values and remove them if found
duplicate_values = diabetes_data.duplicated().sum()
print("\nDuplicate values:", duplicate_values)
# Remove duplicate rows
diabetes_data = diabetes_data.drop_duplicates()
# Display updated information
print("\nUpdated number of records:", len(diabetes_data))
```

```
Duplicate values: 0

Updated number of records: 768
```

jupyter task no 6.1 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [13]:
```
#Task no- 2
import pandas as pd
import matplotlib.pyplot as plt
# 1. Import the diabetes dataset
diabetes_data = pd.read_csv('diabetes.csv')
```

In [14]:
```
# 2. Read the top 5 records from the dataset
top_5_records = diabetes_data.head(5)
print("Top 5 records:")
print(top_5_records)
```

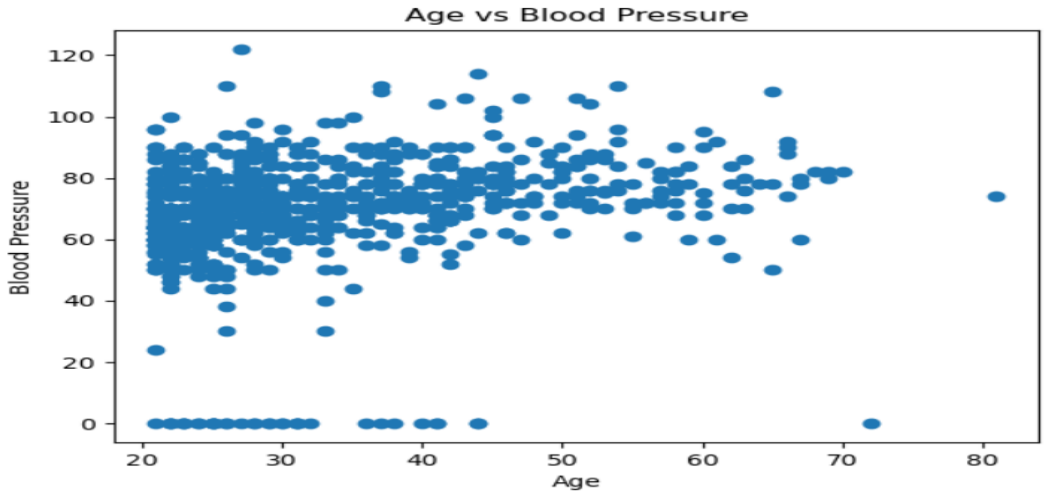
```
Top 5 records:
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0            6      148             72             35        0  33.6
1            1       85             66             29        0  26.6
2            8      183             64              0        0  23.3
3            1       89             66             23       94  28.1
4            0      137             40             35      168  43.1

   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1
```

jupyter task no 6.1 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [15]:
```
# 3. Generate scatter plot for age vs blood pressure
plt.scatter(diabetes_data['Age'], diabetes_data['BloodPressure'])
plt.xlabel('Age')
plt.ylabel('Blood Pressure')
plt.title('Age vs Blood Pressure')
plt.show()
```

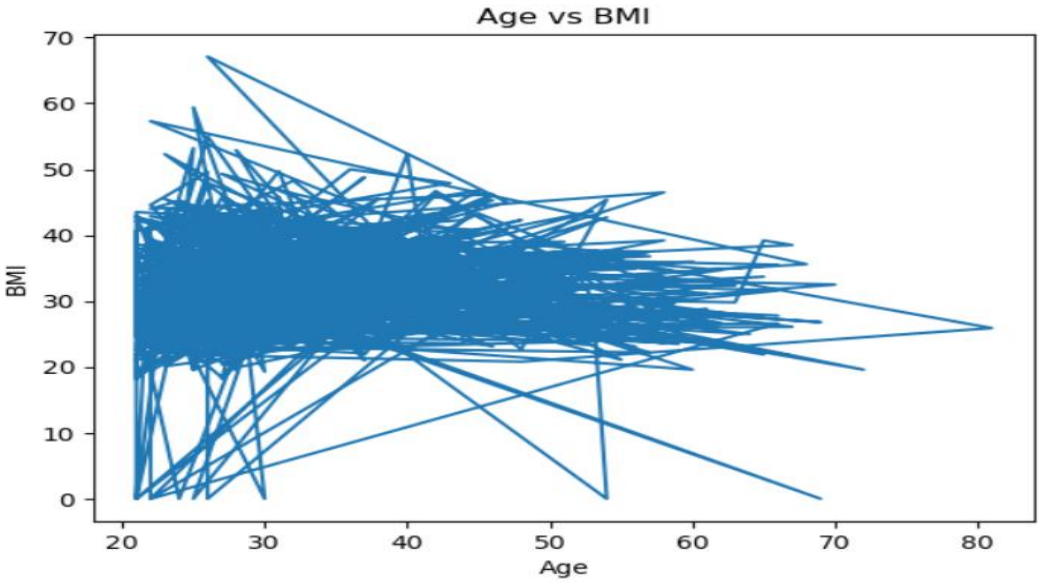


jupyter task no 6.1 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [16]:
```
# 4. Generate plot for age vs BMI
plt.plot(diabetes_data['Age'], diabetes_data['BMI'])
plt.xlabel('Age')
plt.ylabel('BMI')
plt.title('Age vs BMI')
plt.show()
```

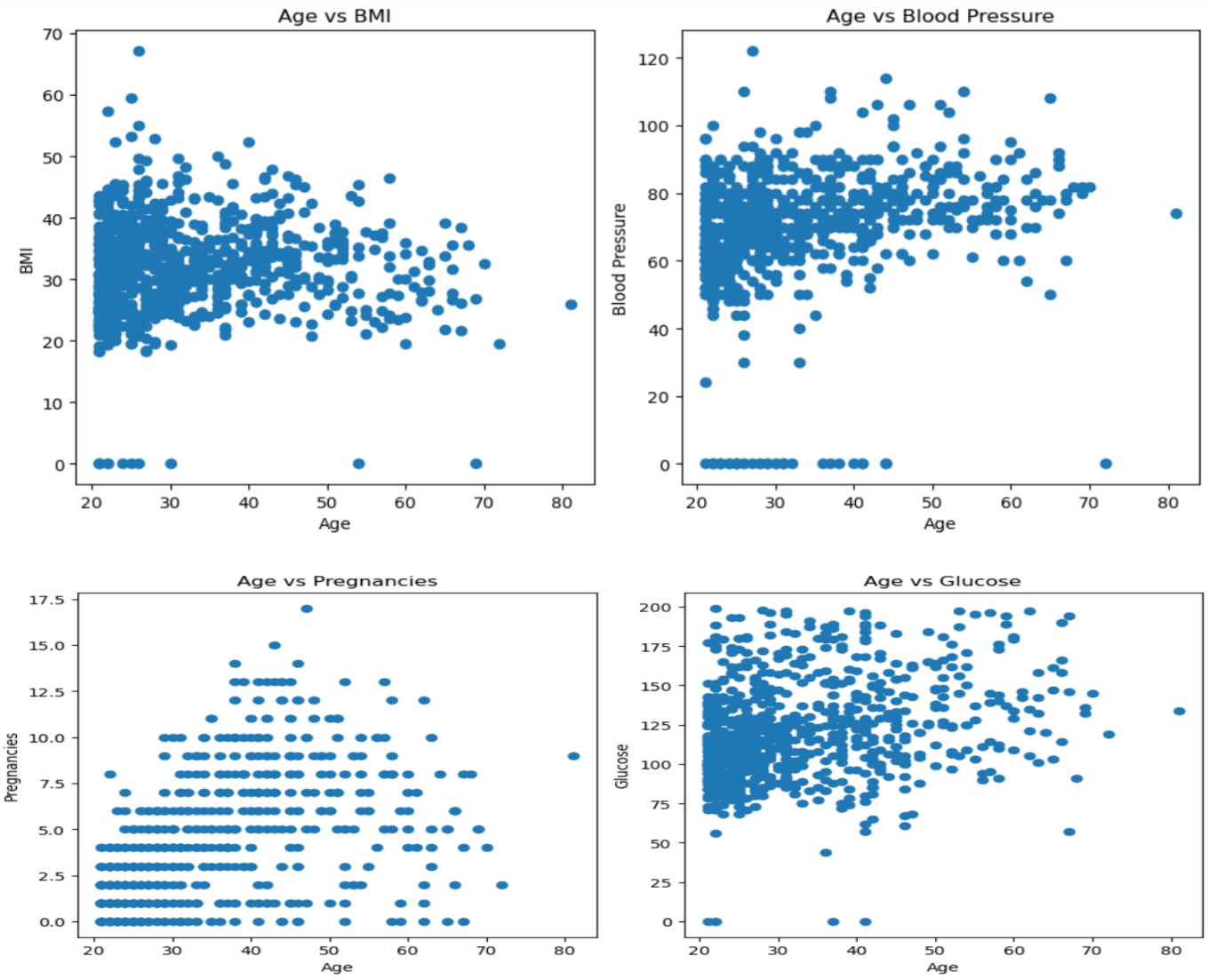


jupyter task no 6.1 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [17]:
```
# 5. Generate 4 scatter plots using subplot
fig, axes = plt.subplots(2, 2, figsize=(10, 10))
axes[0, 0].scatter(diabetes_data['Age'], diabetes_data['BMI'])
axes[0, 0].set_xlabel('Age')
axes[0, 0].set_ylabel('BMI')
axes[0, 0].set_title('Age vs BMI')
axes[0, 1].scatter(diabetes_data['Age'], diabetes_data['BloodPressure'])
axes[0, 1].set_xlabel('Age')
axes[0, 1].set_ylabel('Blood Pressure')
axes[0, 1].set_title('Age vs Blood Pressure')
axes[1, 0].scatter(diabetes_data['Age'], diabetes_data['Pregnancies'])
axes[1, 0].set_xlabel('Age')
axes[1, 0].set_ylabel('Pregnancies')
axes[1, 0].set_title('Age vs Pregnancies')
axes[1, 1].scatter(diabetes_data['Age'], diabetes_data['Glucose'])
axes[1, 1].set_xlabel('Age')
axes[1, 1].set_ylabel('Glucose')
axes[1, 1].set_title('Age vs Glucose')
plt.tight_layout()
plt.show()
```

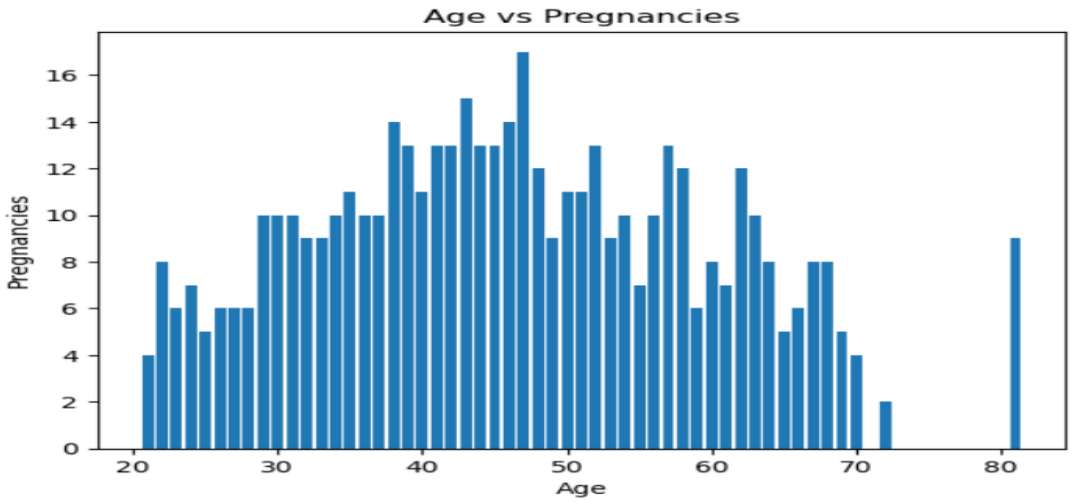


jupyter task no 6.1 Last Checkpoint: an hour ago (autosaved)

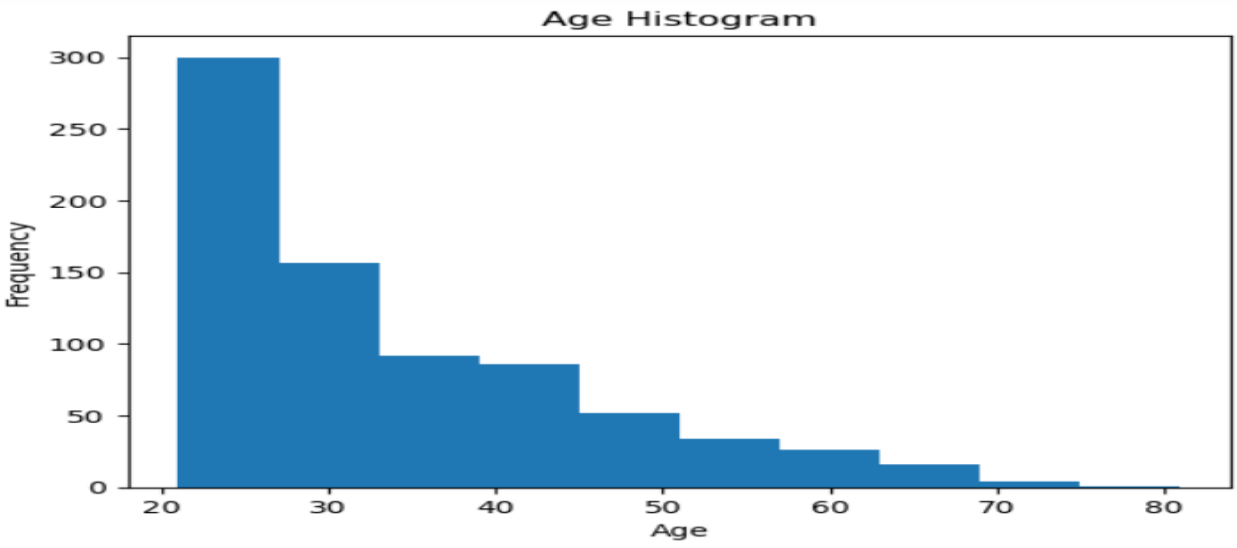
File Edit View Insert Cell Kernel Widgets Help

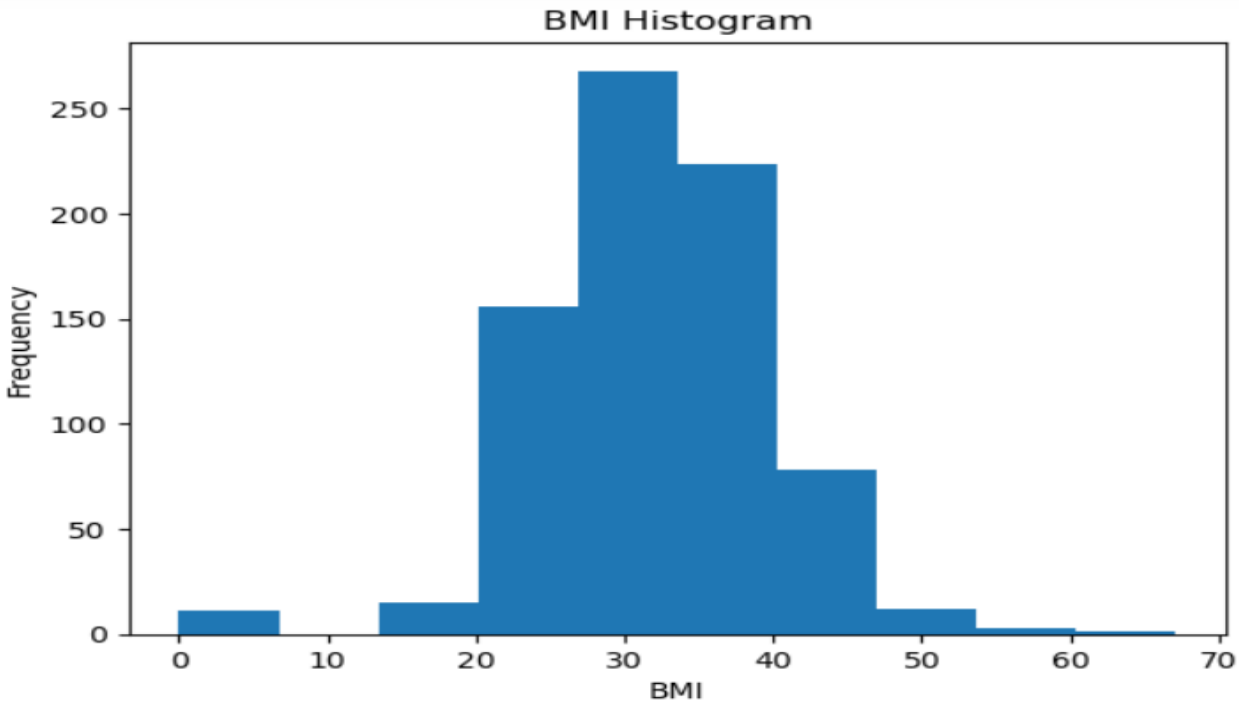
Run Code

In [18]:
```
# 6. Generate bar plot for age vs pregnancy
plt.bar(diabetes_data['Age'], diabetes_data['Pregnancies'])
plt.xlabel('Age')
plt.ylabel('Pregnancies')
plt.title('Age vs Pregnancies')
plt.show()
```

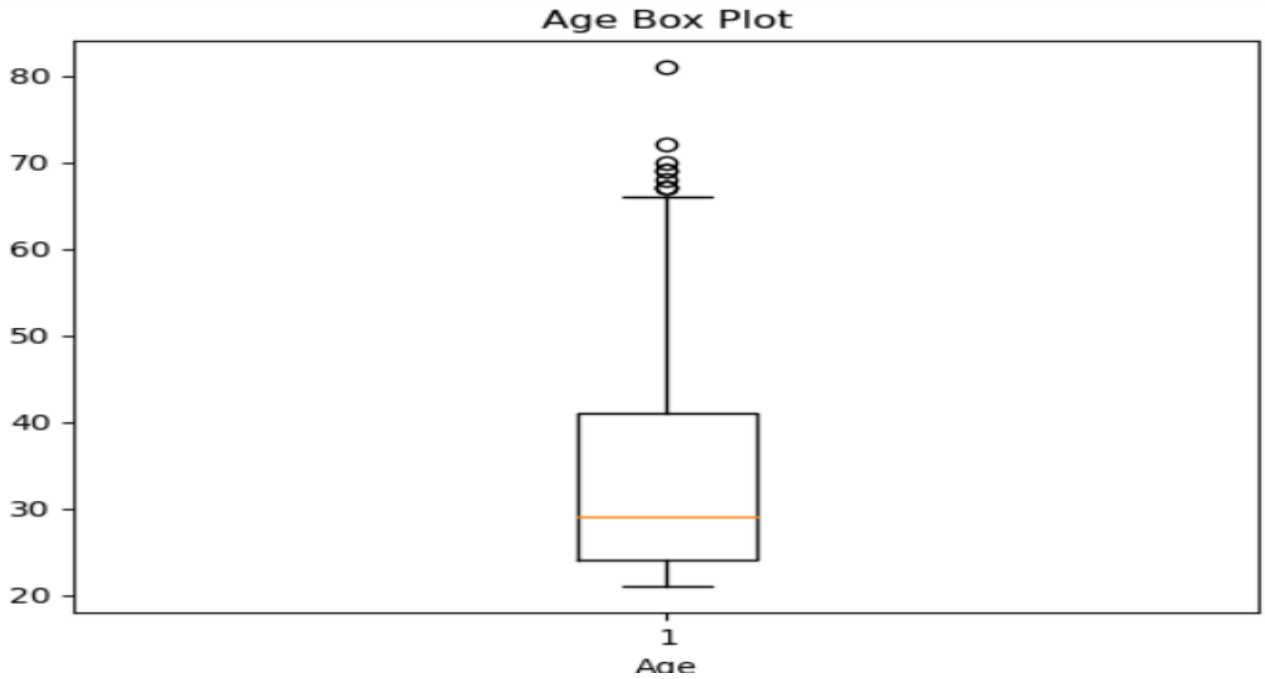


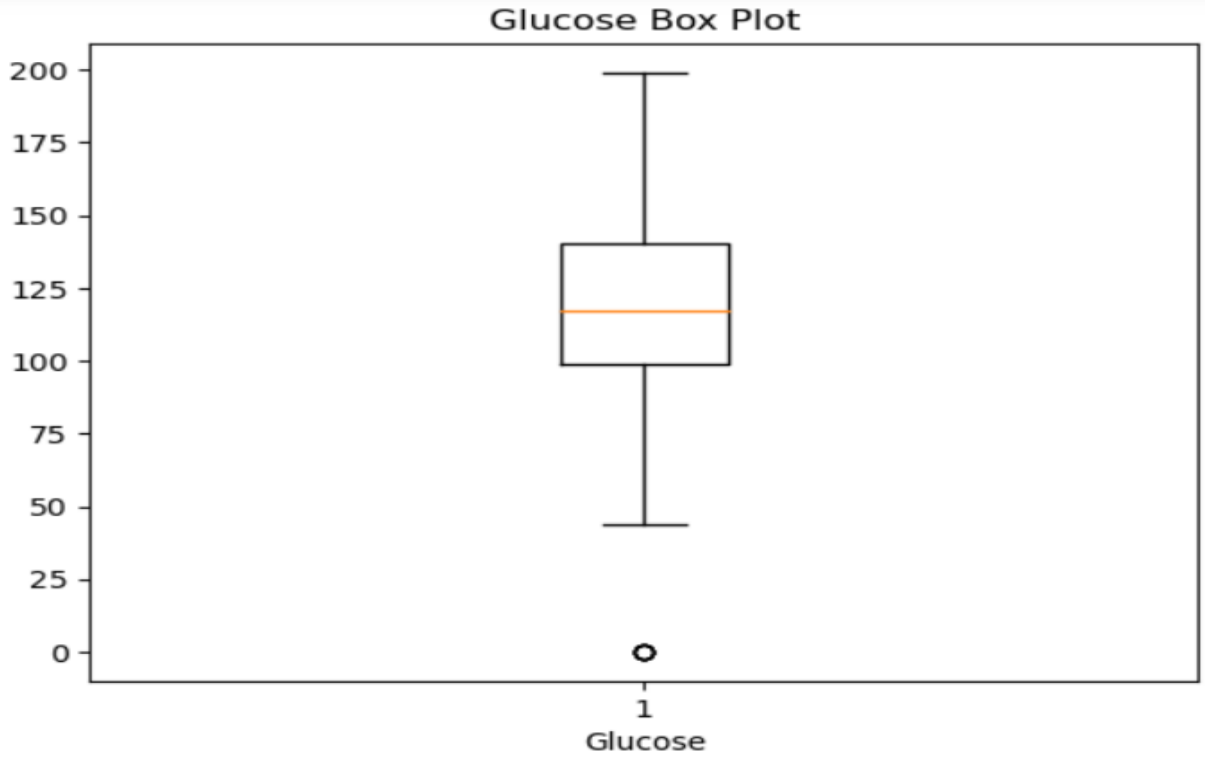
In [19]:
```
# 7. Generate histogram for age and BMI
plt.hist(diabetes_data['Age'], bins=10)
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.title('Age Histogram')
plt.show()
plt.hist(diabetes_data['BMI'], bins=10)
plt.xlabel('BMI')
plt.ylabel('Frequency')
plt.title('BMI Histogram')
plt.show()
```





In [20]:
```
# 8. Generate box plot for age and glucose
plt.boxplot(diabetes_data['Age'])
plt.xlabel('Age')
plt.title('Age Box Plot')
plt.show()
plt.boxplot(diabetes_data['Glucose'])
plt.xlabel('Glucose')
plt.title('Glucose Box Plot')
plt.show()
```





In [21]:
```
# 9. Generate a pie chart for the outcome variable in the diabetes dataset
outcome_counts = diabetes_data['Outcome'].value_counts()
labels = ['No Diabetes', 'Diabetes']
plt.pie(outcome_counts, labels=labels, autopct='%1.1f%%', startangle=90)
plt.axis('equal')
plt.title('Diabetes Outcome')
plt.show()
```