Name: Muhammad Ihtesham Yousaf

RegN0: CS120202069

Sem: 6th Section B

In [ ]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import math
```

In [4]:
```
data = pd.read_csv('diabetes.csv')
```

In [6]:
```
data.head(10)
```

Out[6]:

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

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

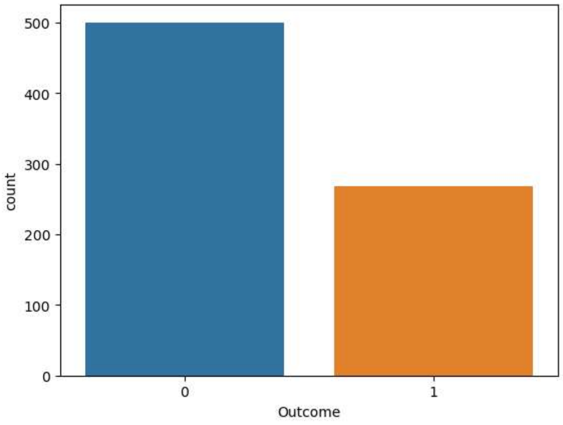
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   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]:
```
print(data.index)
print((len(data.index)))
```

```
RangeIndex(start=0, stop=768, step=1)
768
```

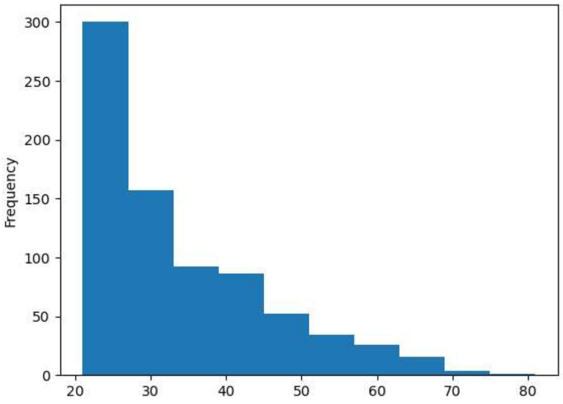
In [9]:
```
sns.countplot(x='Outcome', data=data)
```

Out[9]:
```
<AxesSubplot:xlabel='Outcome', ylabel='count'>
```



In [11]:
```
data['Age'].plot.hist()
```

Out[11]:
```
<AxesSubplot:ylabel='Frequency'>
```



In [12]:
```
data.isnull()
```

Out[12]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunct
0	False	False	False	False	False	False	Fa
1	False	False	False	False	False	False	Fa
2	False	False	False	False	False	False	Fa
3	False	False	False	False	False	False	Fa
4	False	False	False	False	False	False	Fa
...	...	...	...	...	...	...	
763	False	False	False	False	False	False	Fa
764	False	False	False	False	False	False	Fa
765	False	False	False	False	False	False	Fa
766	False	False	False	False	False	False	Fa
767	False	False	False	False	False	False	Fa

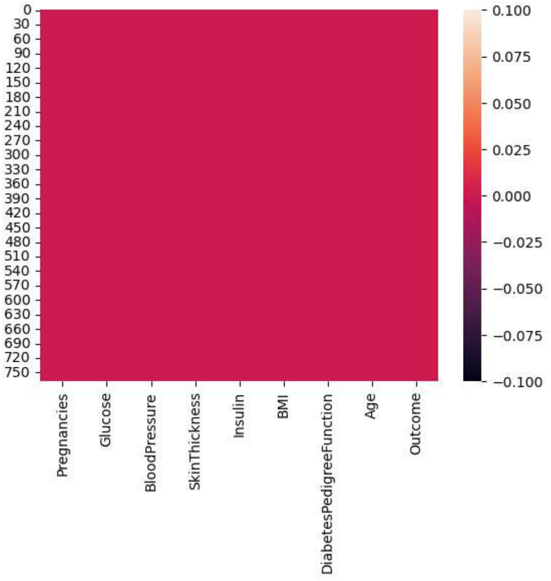
768 rows × 9 columns

In [13]:
```
data.isnull().sum()
```

Out[13]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [14]:
```
sns.heatmap(data.isnull())
```

Out[14]:
```
<AxesSubplot:>
```



```
In [15]: data.dropna(inplace=True)
```

```
In [19]: from sklearn.model_selection import train_test_split
```

```
In [21]: X = data.drop(['Outcome'], axis = 1)
         y = data['Outcome']
```

```
In [22]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, rando
```

```
In [23]: from sklearn.linear_model import LogisticRegression
```

```
In [24]: classifier = LogisticRegression()
```

```
         classifier.fit(X_train, y_train)
```

```
In [26]: prediction = classifier.predict(X_test)
```

```
In [27]: print(prediction)

         [0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 1 0 1 0
          0 0 1 0 0 0 0 0 0 1 1 0 0 0 0 1 0 1 0 1 0 1 0 1 0 1 0 0 0 0 0 1 1 1 1 1 0
          1 0 1 0 0 1 1 0 0 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 1 0 0 0
          0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 1 1 0 0 0
          0 0 0 1 0 0]
```

```
In [28]: from sklearn.metrics import accuracy_score
```

```
In [29]: accuracy_score(y_test, prediction)

Out[29]: 0.7792207792207793
```

```
In [30]: from sklearn.metrics import classification_report
```

```
In [33]: print(classification_report(y_test, prediction))

                       precision    recall  f1-score   support

                    0       0.79      0.90      0.84        99
                    1       0.76      0.56      0.65        55

             accuracy                           0.78       154
            macro avg       0.77      0.73      0.74       154
         weighted avg       0.78      0.78      0.77       154
```

```
In [34]: from sklearn.metrics import confusion_matrix
```

```
In [35]: print(confusion_matrix(y_test, prediction))

         [[89 10]
          [24 31]]
```