In [1]:

```
#Step 1: Import Libraries
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns
import missingno as msno
from scipy import stats
import statsmodels.api as sm
from scipy.stats import chi2_contingency
from scipy.stats import linregress
```

In [2]:

```
#Step 2: load dataset
data = pd.read_csv('C:/Users/ericy/Desktop/medical_clean.csv')
```

In [3]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 50 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   CaseOrder           10000 non-null  int64  
 1   Customer_id         10000 non-null  object 
 2   Interaction         10000 non-null  object 
 3   UID                 10000 non-null  object 
 4   City                10000 non-null  object 
 5   State               10000 non-null  object 
 6   County              10000 non-null  object 
 7   Zip                 10000 non-null  int64  
 8   Lat                 10000 non-null  float64
 9   Lng                 10000 non-null  float64
 10  Population          10000 non-null  int64  
 11  Area                10000 non-null  object 
 12  TimeZone            10000 non-null  object 
 13  Job                 10000 non-null  object 
 14  Children            10000 non-null  int64  
 15  Age                 10000 non-null  int64  
 16  Income              10000 non-null  float64
 17  Marital             10000 non-null  object 
 18  Gender              10000 non-null  object 
 19  ReAdmis             10000 non-null  object 
 20  VitD_levels         10000 non-null  float64
 21  Doc_visits          10000 non-null  int64  
 22  Full_meals_eaten    10000 non-null  int64  
 23  vitD_supp           10000 non-null  int64  
 24  Soft_drink          10000 non-null  object 
 25  Initial_admin       10000 non-null  object 
 26  HighBlood           10000 non-null  object 
 27  Stroke              10000 non-null  object 
 28  Complication_risk   10000 non-null  object 
 29  Overweight          10000 non-null  object 
 30  Arthritis           10000 non-null  object 
 31  Diabetes            10000 non-null  object 
 32  Hyperlipidemia      10000 non-null  object 
 33  BackPain            10000 non-null  object 
 34  Anxiety             10000 non-null  object 
 35  Allergic_rhinitis   10000 non-null  object 
 36  Reflux_esophagitis  10000 non-null  object 
 37  Asthma              10000 non-null  object 
 38  Services            10000 non-null  object 
 39  Initial_days        10000 non-null  float64
 40  TotalCharge         10000 non-null  float64
 41  Additional_charges  10000 non-null  float64
 42  Item1               10000 non-null  int64  
 43  Item2               10000 non-null  int64  
 44  Item3               10000 non-null  int64  
 45  Item4               10000 non-null  int64  
 46  Item5               10000 non-null  int64  
 47  Item6               10000 non-null  int64  
 48  Item7               10000 non-null  int64  
 49  Item8               10000 non-null  int64  
dtypes: float64(7), int64(16), object(27)
memory usage: 3.8+ MB
```

In [4]:

```
# C: Univariate Statistics.  2 continuous & 2 categorical
#Univariate 2 categorical: Stroke and Doc_visits
#Univariate Statistical Analysis of 'Stroke' using Frequency Table
data['Stroke'].value_counts()
```

Out[4]:

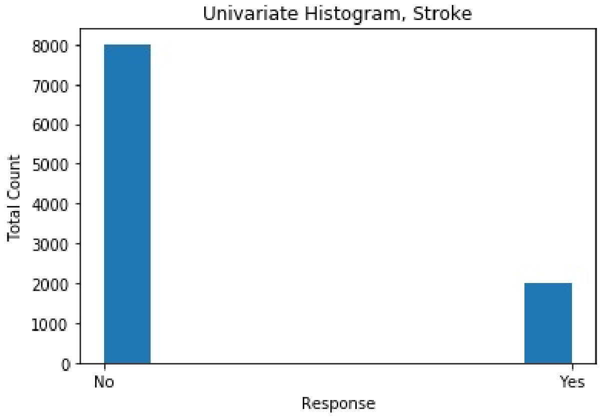
```
No     8007
Yes    1993
Name: Stroke, dtype: int64
```

In [5]:

```
# Visualization of 'Stroke' using histogram
plt.hist(data['Stroke'])
plt.title('Univariate Histogram, Stroke')
plt.xlabel('Response')
plt.ylabel('Total Count')
```

Out[5]:

```
Text(0, 0.5, 'Total Count')
```



In [6]:

```
#Initial_admin univariate statistical analysis with frequency table
data['Initial_admin'].value_counts()
```

Out[6]:

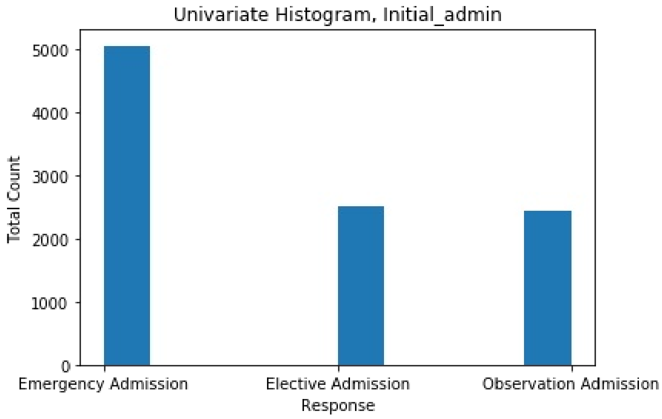
```
Emergency Admission      5060
Elective Admission       2504
Observation Admission    2436
Name: Initial_admin, dtype: int64
```

In [7]:

```
#Initial_admin visualization using histogram
plt.hist(data['Initial_admin'])
plt.title('Univariate Histogram, Initial_admin')
plt.xlabel('Response')
plt.ylabel('Total Count')
```

Out[7]:

```
Text(0, 0.5, 'Total Count')
```



In [8]:

```
#2 continuous variables using Univariate Statistical Analysis - Income & TotalCharge
#Summary Stats of 'Income'
data['Income'].mean()
```

Out[8]:

```
40490.495159999846
```

In [9]:

```
data['Income'].median()
```

Out[9]:

```
33768.42
```

In [10]:

```
data['Income'].std()
```

Out[10]:

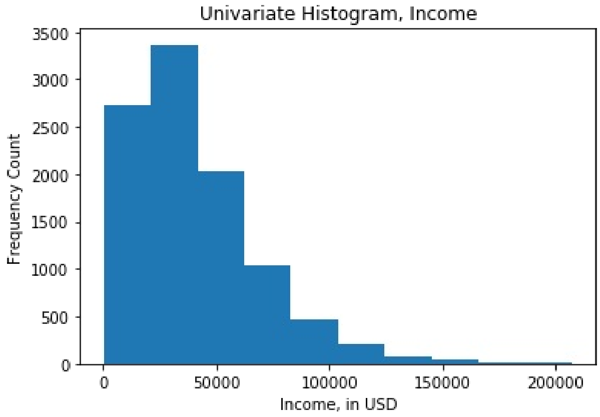
```
28521.15329318396
```

In [11]:

```
#'Income' Visualization
plt.hist(data['Income'])
plt.title('Univariate Histogram, Income')
plt.xlabel('Income, in USD')
plt.ylabel('Frequency Count')
```

Out[11]:

```
Text(0, 0.5, 'Frequency Count')
```



In [12]:

```
#Summary Stats of 'TotalCharge'
data['TotalCharge'].mean()
```

Out[12]:

```
5312.172768750177
```

In [13]:

```
data['TotalCharge'].median()
```

Out[13]:

```
5213.951999999999
```

In [14]:

```
data['TotalCharge'].std()
```

Out[14]:

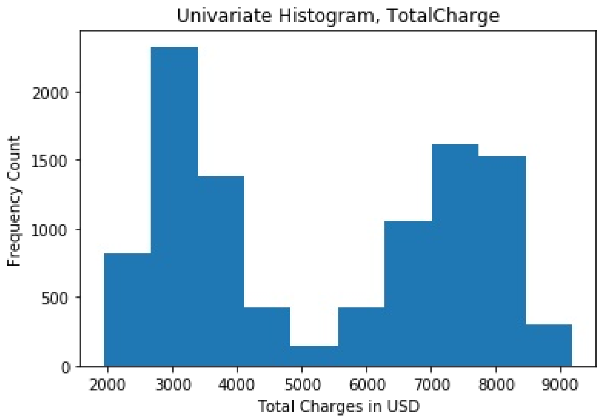
```
2180.3938378109415
```

In [15]:

```
#'TotalCharge' Visualization
plt.hist(data['TotalCharge'])
plt.title('Univariate Histogram, TotalCharge')
plt.xlabel('Total Charges in USD')
plt.ylabel('Frequency Count')
```

Out[15]:

```
Text(0, 0.5, 'Frequency Count')
```



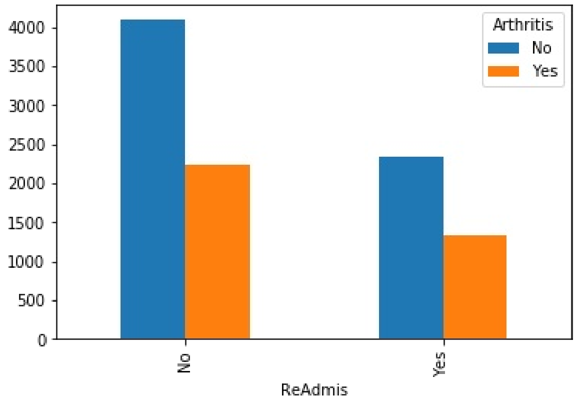
In [16]:

```
# D: Bivariate Statistics: 2 continuous & 2 categorical
# 2 categorical: Variables ReAdmis and Arthritis using Chi Square Analyis
# Visualization
#Arthritis
ReArN = pd.crosstab(data['ReAdmis'], data['Arthritis'])
print(ReArN)
ReArN.plot(kind='bar')
```

```
Arthritis    No   Yes
ReAdmis
No         4086  2245
Yes        2340  1329
```

Out[16]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x16863b1ed48>
```



In [17]:

```
# Bivariate Statistical Analysis of ReAdmis and Arthritis using Chi Squared
chi2ar, par, dofar, expectedar = chi2_contingency(ReArN)
print('chi2 stat', chi2ar, 'p value', par, 'dof', dofar, 'expected values', expectedar)
if par < .1:
    print('Reject the Null Hypothesis')
else:
    print('Accept the Null Hypothesis')
```

```
chi2 stat 0.5545124468934712 p value 0.4564797501244029 dof 1 expected values [[4068.3006 2262.6994]
 [2357.6994 1311.3006]]
Accept the Null Hypothesis
```

In [18]:

```
# 2 Continuous variables: Additional_charges and Initial_days
# Bivariate Statistical Analysis of Income and Age using Regression
from scipy.stats import linregress
slope, intercept, r_value, p_value, std_err = stats.linregress(data['Additional_charges'],data['Initial_days'])
print('p_value is', p_value, 'r-squared', r_value**2)
```

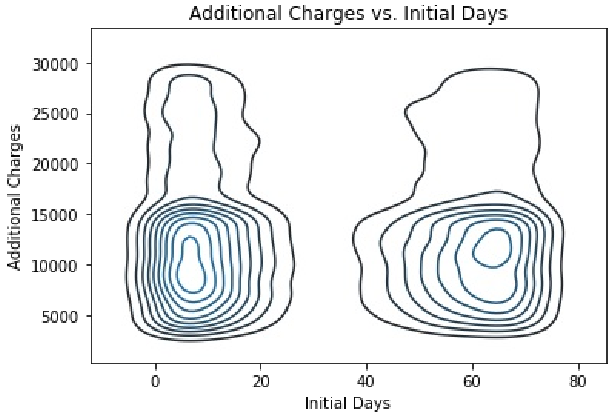
```
p_value is 0.6593322230998617 r-squared 1.9438248404066487e-05
```

In [19]:

```
# Citation for Kernal Density Estimate graph usage: (seaborn.kdeplot.  N.d.)
#Kernel Density Estimate KDE plot two continuous variables
sns.kdeplot(data['Initial_days'], data['Additional_charges'])
plt.title('Additional Charges vs. Initial Days')
plt.xlabel('Initial Days')
plt.ylabel('Additional Charges')
```

Out[19]:

```
Text(0, 0.5, 'Additional Charges')
```


Additional Charges vs. Initial Days
Additional Charges
Initial Days