In [2]:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Loading dataset

In [5]:

```python
df=pd.read_excel("DoctorVisits.xlsx")
```

In [6]:

```python
df
```

Out[6]:

	Unnamed: 0	visits	gender	age	income	illness	reduced	health	private	freepoor	freerepat	nchronic	lchronic
0	1	1	female	0.19	0.55	1	4	1	yes	no	no	no	no
1	2	1	female	0.19	0.45	1	2	1	yes	no	no	no	no
2	3	1	male	0.19	0.90	3	0	0	no	no	no	no	no
3	4	1	male	0.19	0.15	1	0	0	no	no	no	no	no
4	5	1	male	0.19	0.45	2	5	1	no	no	no	yes	no
...	...	...	...	...	...	...	...	...	...	...	...	...	...
5185	5186	0	female	0.22	0.55	0	0	0	no	no	no	no	no
5186	5187	0	male	0.27	1.30	0	0	1	no	no	no	no	no
5187	5188	0	female	0.37	0.25	1	0	1	no	no	yes	no	no
5188	5189	0	female	0.52	0.65	0	0	0	no	no	no	no	no
5189	5190	0	male	0.72	0.25	0	0	0	no	no	yes	no	no

5190 rows × 13 columns

In [5]:

```python
df.head(10)
```

Out[5]:

	Unnamed: 0	visits	gender	age	income	illness	reduced	health	private	freepoor	freerepat	nchronic	lchronic
0	1	1	female	0.19	0.55	1	4	1	yes	no	no	no	no
1	2	1	female	0.19	0.45	1	2	1	yes	no	no	no	no
2	3	1	male	0.19	0.90	3	0	0	no	no	no	no	no
3	4	1	male	0.19	0.15	1	0	0	no	no	no	no	no
4	5	1	male	0.19	0.45	2	5	1	no	no	no	yes	no
5	6	1	female	0.19	0.35	5	1	9	no	no	no	yes	no
6	7	1	female	0.19	0.55	4	0	2	no	no	no	no	no
7	8	1	female	0.19	0.15	3	0	6	no	no	no	no	no
8	9	1	female	0.19	0.65	2	0	5	yes	no	no	no	no
9	10	1	male	0.19	0.15	1	0	0	yes	no	no	no	no

In [6]:

```python
df.tail(10)
```

Out[6]:

	Unnamed: 0	visits	gender	age	income	illness	reduced	health	private	freepoor	freerepat	nchronic	lchronic
5180	5181	0	male	0.19	0.25	1	0	1	no	yes	no	no	no
5181	5182	0	male	0.19	0.75	1	0	0	no	no	no	no	no
5182	5183	0	female	0.19	0.45	0	0	0	no	no	no	no	no
5183	5184	0	male	0.22	1.10	0	0	0	no	no	no	no	no
5184	5185	0	male	0.22	1.50	0	0	0	no	no	no	no	no
5185	5186	0	female	0.22	0.55	0	0	0	no	no	no	no	no
5186	5187	0	male	0.27	1.30	0	0	1	no	no	no	no	no
5187	5188	0	female	0.37	0.25	1	0	1	no	no	yes	no	no
5188	5189	0	female	0.52	0.65	0	0	0	no	no	no	no	no
5189	5190	0	male	0.72	0.25	0	0	0	no	no	yes	no	no

Information of data set

In [7]:

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5190 entries, 0 to 5189
Data columns (total 13 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Unnamed: 0  5190 non-null   int64
 1   visits      5190 non-null   int64
 2   gender      5190 non-null   object
 3   age         5190 non-null   float64
 4   income      5190 non-null   float64
 5   illness     5190 non-null   int64
 6   reduced     5190 non-null   int64
 7   health      5190 non-null   int64
 8   private     5190 non-null   object
 9   freepoor    5190 non-null   object
 10  freerepat   5190 non-null   object
 11  nchronic    5190 non-null   object
 12  lchronic    5190 non-null   object
dtypes: float64(2), int64(5), object(6)
memory usage: 527.2+ KB
```

Checking data contain null values or not

In [8]:

```python
df.isnull().sum()
```

Out[8]:

```
Unnamed: 0    0
visits        0
gender        0
age           0
income        0
illness       0
reduced       0
health        0
private       0
freepoor      0
freerepat     0
nchronic      0
lchronic      0
dtype: int64
```

In [9]:

```python
#  No Null values are there in the given data set
```

Accessing columns

In [7]:

```python
df["illness"]
```

Out[7]:

```
0       1
1       1
2       3
3       1
4       2
       ..
5185    0
5186    0
5187    1
5188    0
5189    0
Name: illness, Length: 5190, dtype: int64
```

Finding the Mean,Min,Max values

In [11]:

```python
df.describe()
```

Out[11]:

	Unnamed: 0	visits	age	income	illness	reduced	health
count	5190.000000	5190.000000	5190.000000	5190.000000	5190.000000	5190.000000	5190.000000
mean	2595.500000	0.301734	0.406385	0.583160	1.431985	0.861850	1.217534
std	1498.368279	0.798134	0.204782	0.368907	1.384152	2.887628	2.124266
min	1.000000	0.000000	0.190000	0.000000	0.000000	0.000000	0.000000
25%	1298.250000	0.000000	0.220000	0.250000	0.000000	0.000000	0.000000
50%	2595.500000	0.000000	0.320000	0.550000	1.000000	0.000000	0.000000
75%	3892.750000	0.000000	0.620000	0.900000	2.000000	0.000000	2.000000
max	5190.000000	9.000000	0.720000	1.500000	5.000000	14.000000	12.000000

Finding total no of people based on their count of illness

In [8]:

```python
df["illness"].value_counts()
```

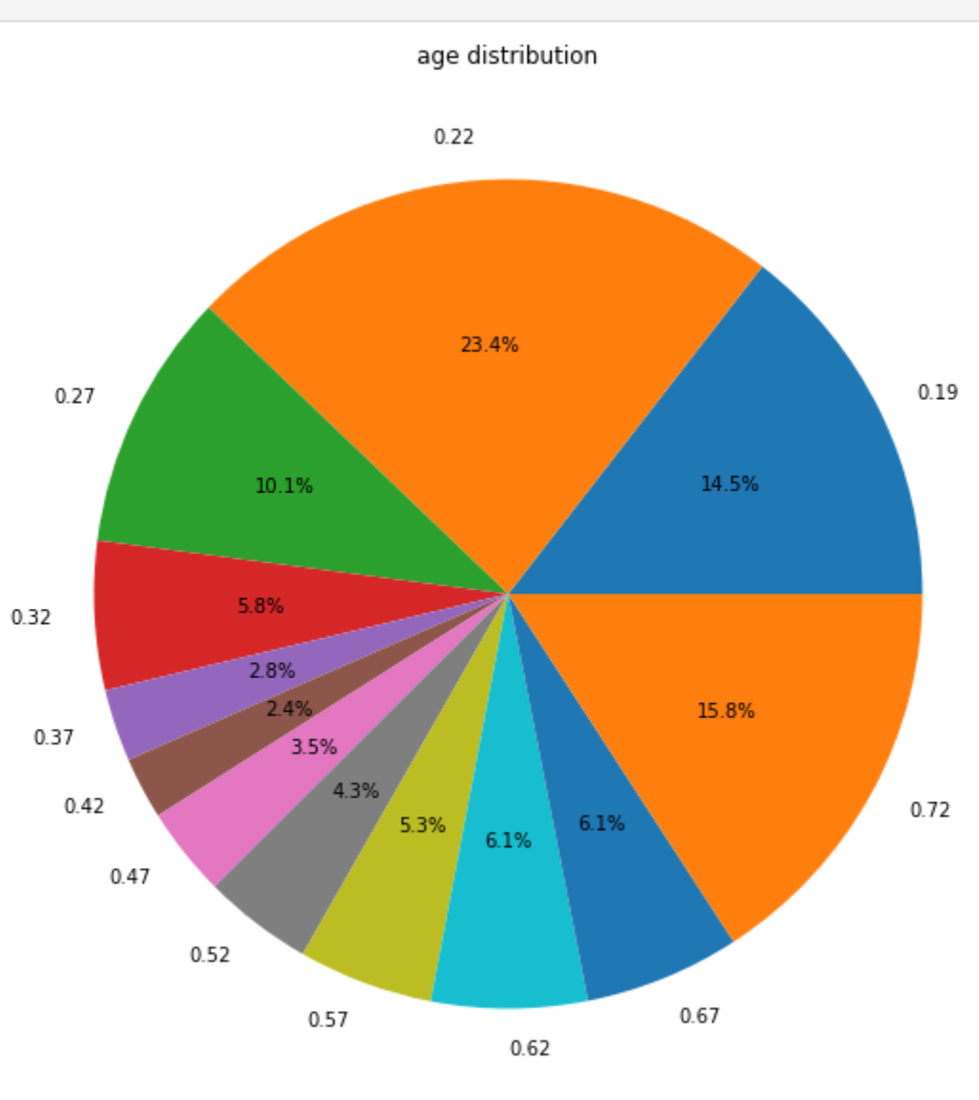
Out[8]:

```
1    1638
0    1554
2     946
3     542
4     274
5     236
Name: illness, dtype: int64
```

Representing percentage of patient and age using pie-chart

In [13]:

```python
age_count=df['age'].value_counts()
age_count=age_count.sort_index()
plt.figure(figsize =(10,10))
plt.pie(age_count, labels=age_count.index, autopct='%1.1f%%')
plt.title('age distribution')
plt.show()
```



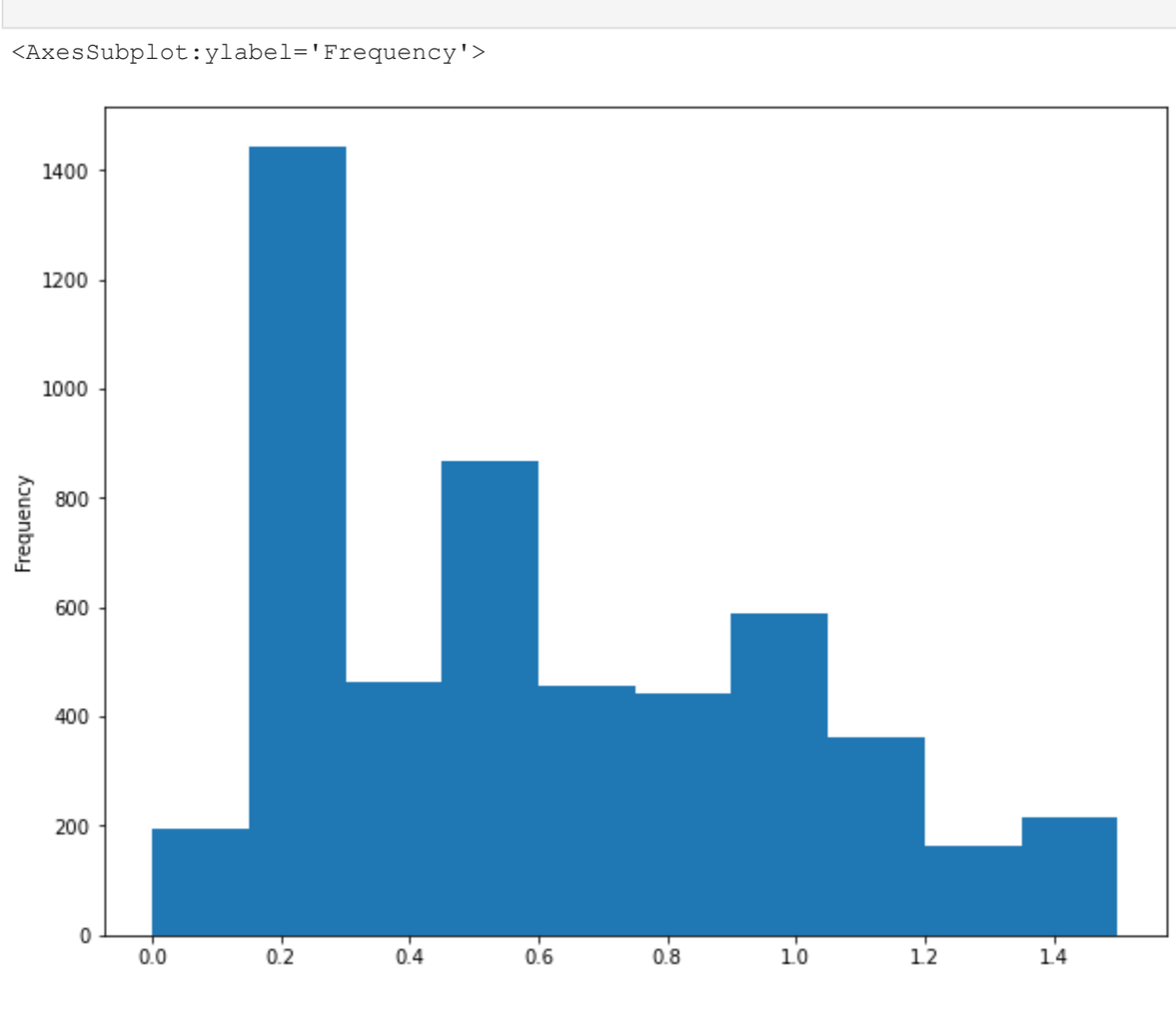
Representing income using histogram

In [14]:

```python
plt.figure(figsize =(10,8))
df["income"].plot(kind='hist')
```

Out[14]:

```
<AxesSubplot:ylabel='Frequency'>
```



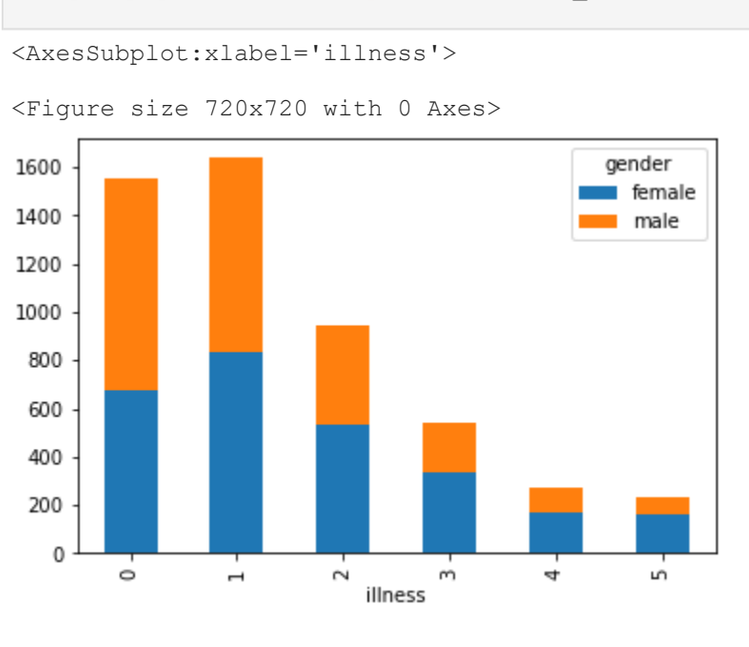
Representing percentage of male and female using piechart

In [75]:

```python
plt.figure(figsize =(10,10))
df.groupby('illness')['gender'].value_counts().unstack().plot(kind='bar', stacked=True)
```

Out[75]:

```
<AxesSubplot:xlabel='illness'>
<Figure size 720x720 with 0 Axes>
```



visualize and analyse the maximum,minimum and medium income

In [42]:

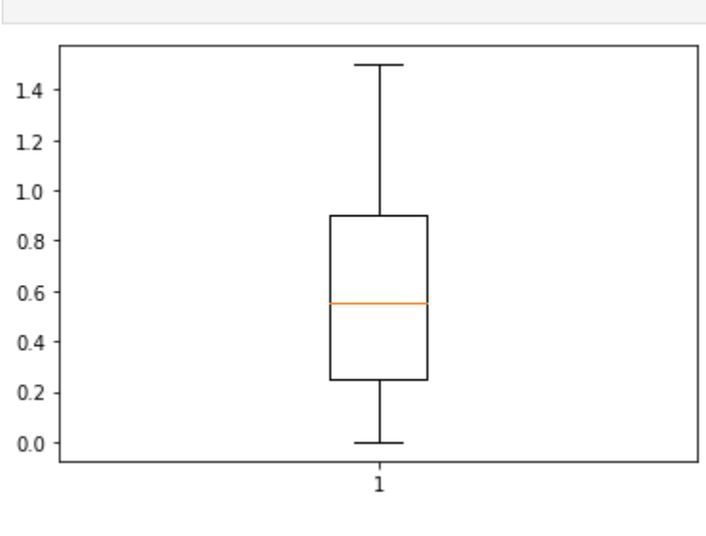
```python
df["income"].describe()
```

Out[42]:

```
count    5190.000000
mean        0.583160
std         0.368907
min         0.000000
25%         0.250000
50%         0.550000
75%         0.900000
max         1.500000
Name: income, dtype: float64
```

In [46]:

```python
a=list(df.income)
plt.boxplot(a)
plt.show()
```



Finding the age of people who are more likely of getting illness

In [45]:

```python
res=dict(df['age'].value_counts())
max(res, key=lambda k: res[k])
```

Out[45]:

```
0.22
```

Finding the no of days of reduced activity of male and female seperately due to illness

In [51]:

```python
df.groupby(['gender','reduced']).mean()
```

Out[51]:

gender	reduced	Unnamed: 0	visits	age	income	illness	health
female	**0**	2524.038512	0.229322	0.465755	0.482735	1.462144	1.115098
	1	1985.768421	0.400000	0.325684	0.542105	2.242105	1.610526
	2	1622.618182	0.672727	0.391455	0.560182	2.236364	1.781818
	3	997.311111	1.333333	0.403111	0.516000	2.733333	1.733333
	4	1237.740741	0.851852	0.458889	0.466667	2.222222	2.074074
	5	1169.055556	1.444444	0.401667	0.614444	2.222222	2.500000
	6	1382.545455	1.363636	0.426364	0.622727	2.363636	1.363636
	7	1034.846154	1.384615	0.436154	0.473462	2.653846	2.230769
	8	1883.090909	1.090909	0.471818	0.404545	2.181818	4.000000
	9	1349.000000	0.500000	0.570000	0.825000	3.000000	1.000000
	10	1099.428571	2.142857	0.512857	0.421429	2.571429	2.000000
	12	1661.000000	2.000000	0.720000	0.250000	3.500000	5.500000
	13	906.000000	4.000000	0.720000	0.300000	4.500000	3.500000
	14	1392.112069	1.543103	0.551724	0.427586	2.534483	4.112069
male	**0**	3008.911019	0.136007	0.344703	0.694398	1.099585	0.924850
	1	2485.158537	0.304878	0.286220	0.676341	1.743902	1.256098
	2	2007.679245	0.471698	0.343585	0.653019	2.358491	1.547170
	3	1909.068966	0.724138	0.334138	0.741379	2.137931	1.689655
	4	1424.000000	0.722222	0.309444	0.869444	2.055556	2.000000
	5	1437.272727	1.136364	0.331818	0.570455	2.272727	2.818182
	6	562.000000	0.833333	0.340000	0.591667	2.500000	2.000000
	7	1716.750000	0.750000	0.314167	0.655000	2.583333	4.333333
	8	680.666667	1.333333	0.365000	0.833333	2.666667	2.000000
	9	1375.400000	2.200000	0.310000	0.392000	2.400000	2.000000
	10	1543.200000	1.800000	0.480000	0.590000	2.600000	4.600000
	11	355.500000	5.000000	0.320000	1.000000	1.500000	0.500000
	12	781.500000	2.000000	0.370000	0.515000	1.500000	1.000000
	13	508.666667	4.000000	0.510000	0.350000	3.333333	2.333333
	14	1236.069444	1.555556	0.476806	0.598611	2.375000	3.527778

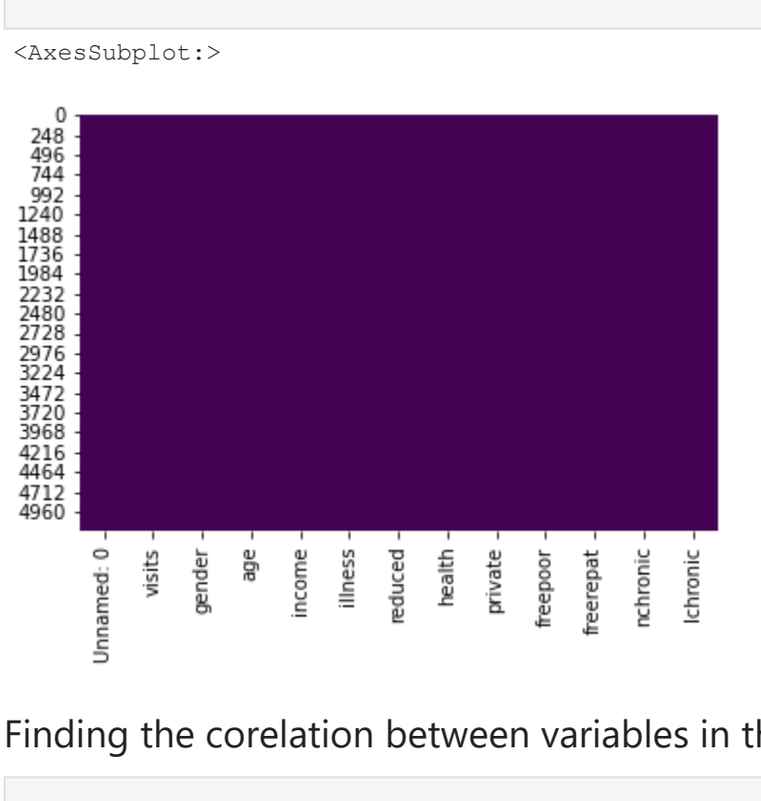
Visualising is any missing values in the dataset based on heat map

In [14]:

```python
sns.heatmap(df.isnull(),cbar=False,cmap='viridis')
```

Out[14]:

```
<AxesSubplot:>
```



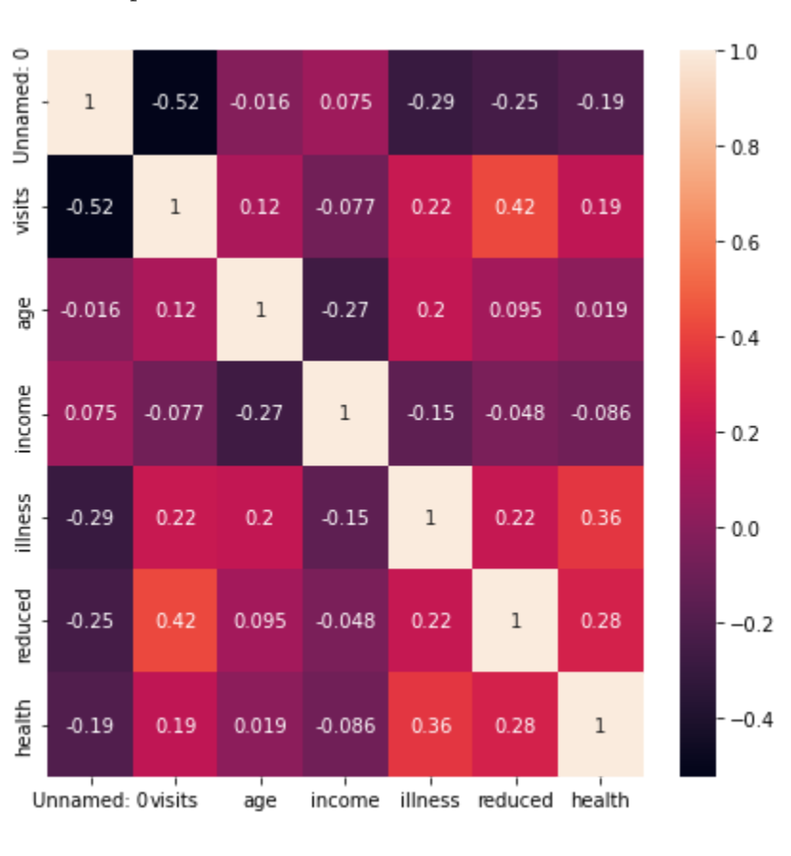
Finding the corelation between variables in the given dataset correlation between different variables

In [54]:

```python
plt.figure(figsize=(7,7))
sns.heatmap(df.corr(),cbar=True,annot=True)
```

Out[54]:

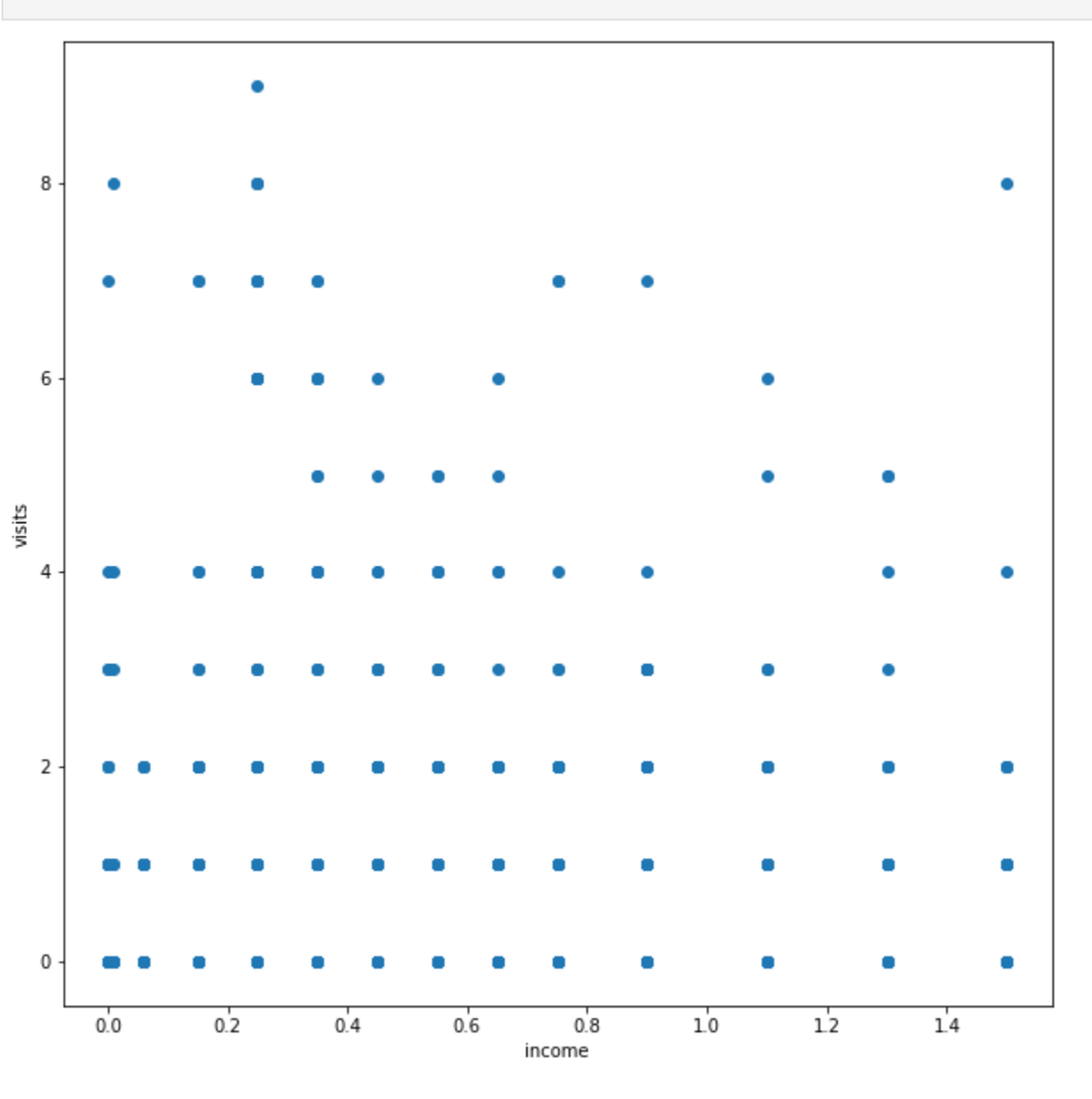
```
<AxesSubplot:>
```



Analyse how the income of a patient affects the no of visits to the hospital

In [55]:

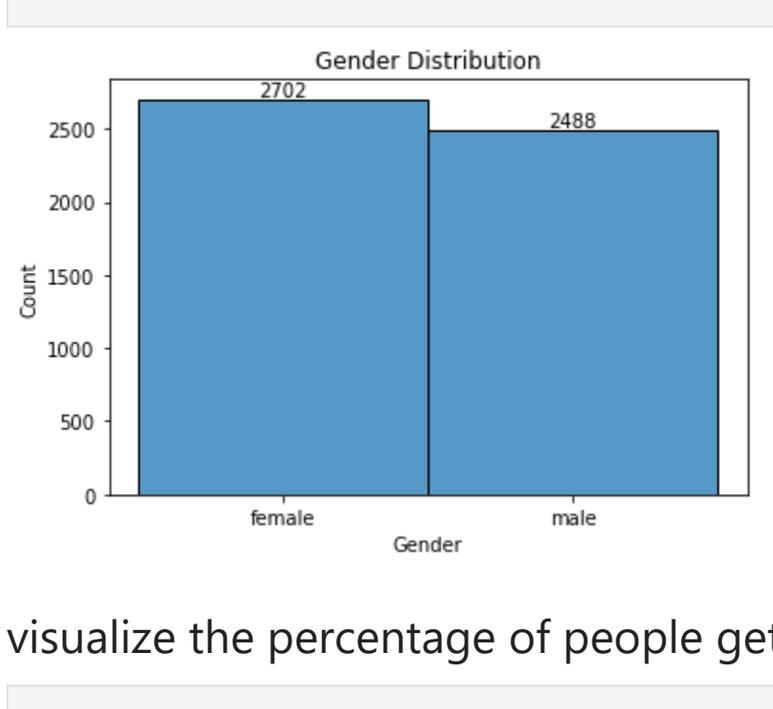
```python
plt.figure(figsize=(10,10))
plt.scatter(x='income',y='visits',data=df)
plt.xlabel('income')
plt.ylabel('visits')
plt.show()
```



visualising the number of males and females affected by illness

In [18]:

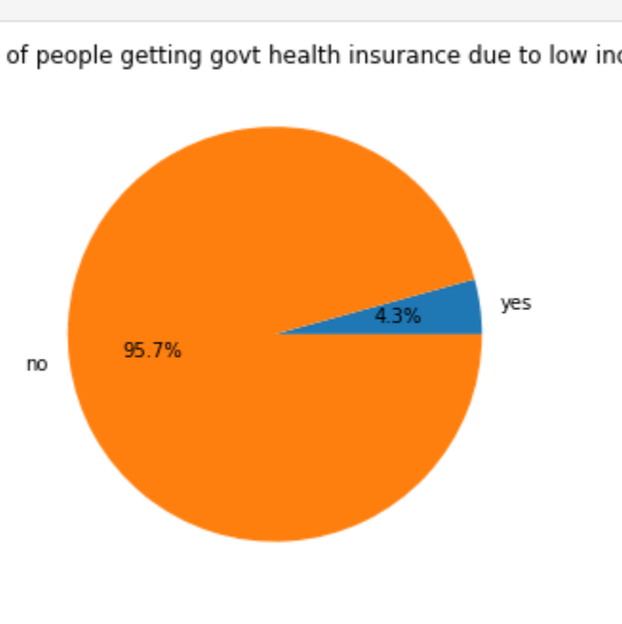
```python
sns.histplot(df['gender'], bins=2)
gender_counts = df['gender'].value_counts()
for i, count in enumerate(gender_counts):
    plt.annotate(str(count), xy=(i, count), ha='center', va='bottom')
plt.xlabel('Gender')
plt.title('Gender Distribution')
plt.show()
```



visualize the percentage of people getting govt health insurance due to low income, due to old age and also the percentage of people having private health insurance

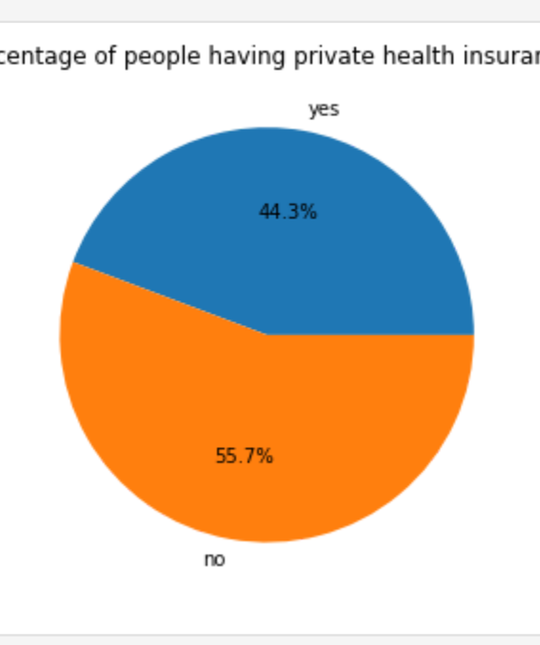
In [82]:

```python
label=['yes','no']
y=df[df['freepoor']=='yes']
n=df[df['freepoor']=='no']
x=[y.shape[0],n.shape[0]]
plt.figure(figsize=(5,5))
plt.pie(x,labels=label, autopct='%1.1f%%')
plt.title('Percentage of people getting govt health insurance due to low income')
plt.show()
```



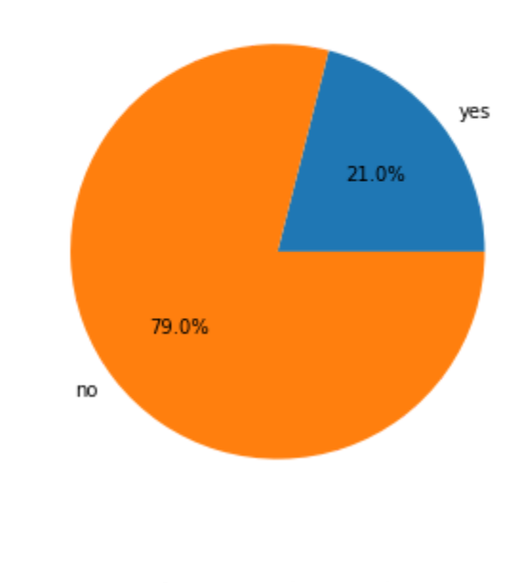
In [83]:

```python
label=['yes','no']
y=df[df['private']=='yes']
n=df[df['private']=='no']
x=[y.shape[0],n.shape[0]]
plt.figure(figsize=(5,5))
plt.pie(x,labels=label,autopct='%1.1f%%')
plt.title('Percentage of people having private health insurance')
plt.show()
```



In [84]:

```python
label=['yes','no']
y=df[df['freerepat']=='yes']
n=df[df['freerepat']=='no']
x=[y.shape[0],n.shape[0]]
plt.figure(figsize=(5,5))
plt.pie(x,labels=label,autopct='%1.1f%%')
plt.title('Percentage of people getting govt insurance due to old age,disability or veteran status')
plt.show()
```



Bar Chart to analyze the days of activity due to illness based on gender

In [19]:

```python
db=df.groupby('gender')['reduced'].sum().to_frame().reset_index()
plt.bar(db['gender'],db['reduced'],color=['red','green'])
plt.title('Bar Chart')
plt.xlabel('gender')
plt.ylabel('reduced activity')
plt.show()
```