In [1]:
```
import pandas as pd
df = pd.read_csv(r"C:/Users/Administrator/Downloads/Heart Disease UCI.csv")
df
```

Out[1]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	tl
0	69	1	0	160	234	1	2	131	0	0.1	1	1	
1	69	0	0	140	239	0	0	151	0	1.8	0	2	
2	66	0	0	150	226	0	0	114	0	2.6	2	0	
3	65	1	0	138	282	1	2	174	0	1.4	1	1	
4	64	1	0	110	211	0	2	144	1	1.8	1	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
292	40	1	3	152	223	0	0	181	0	0.0	0	0	
293	39	1	3	118	219	0	0	140	0	1.2	1	0	
294	35	1	3	120	198	0	0	130	1	1.6	1	0	
295	35	0	3	138	183	0	0	182	0	1.4	0	0	
296	35	1	3	126	282	0	2	156	1	0.0	0	0	

297 rows × 14 columns

In [53]:
```
# To Display 10 rows
df.head(10)

# check data types
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 297 entries, 0 to 296
Data columns (total 16 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   age         297 non-null    int64  
 1   sex         297 non-null    int64  
 2   cp          297 non-null    int64  
 3   trestbps    297 non-null    int64  
 4   chol        297 non-null    int64  
 5   fbs         297 non-null    int64  
 6   restecg     297 non-null    int64  
 7   thalach     297 non-null    int64  
 8   exang       297 non-null    int64  
 9   oldpeak     297 non-null    float64
 10  slope       297 non-null    int64  
 11  ca          297 non-null    int64  
 12  thal        297 non-null    int64  
 13  condition   297 non-null    int64  
 14  risk_score  297 non-null    float64
 15  risk_level  297 non-null    object 
dtypes: float64(2), int64(13), object(1)
memory usage: 37.3+ KB
```

In [54]:
```
# missing values
df.isnull().sum()
```

Out[54]:
```
age          0
sex          0
cp           0
trestbps     0
chol         0
fbs          0
restecg      0
thalach      0
exang        0
oldpeak      0
slope        0
ca           0
thal         0
condition    0
risk_score   0
risk_level   0
dtype: int64
```

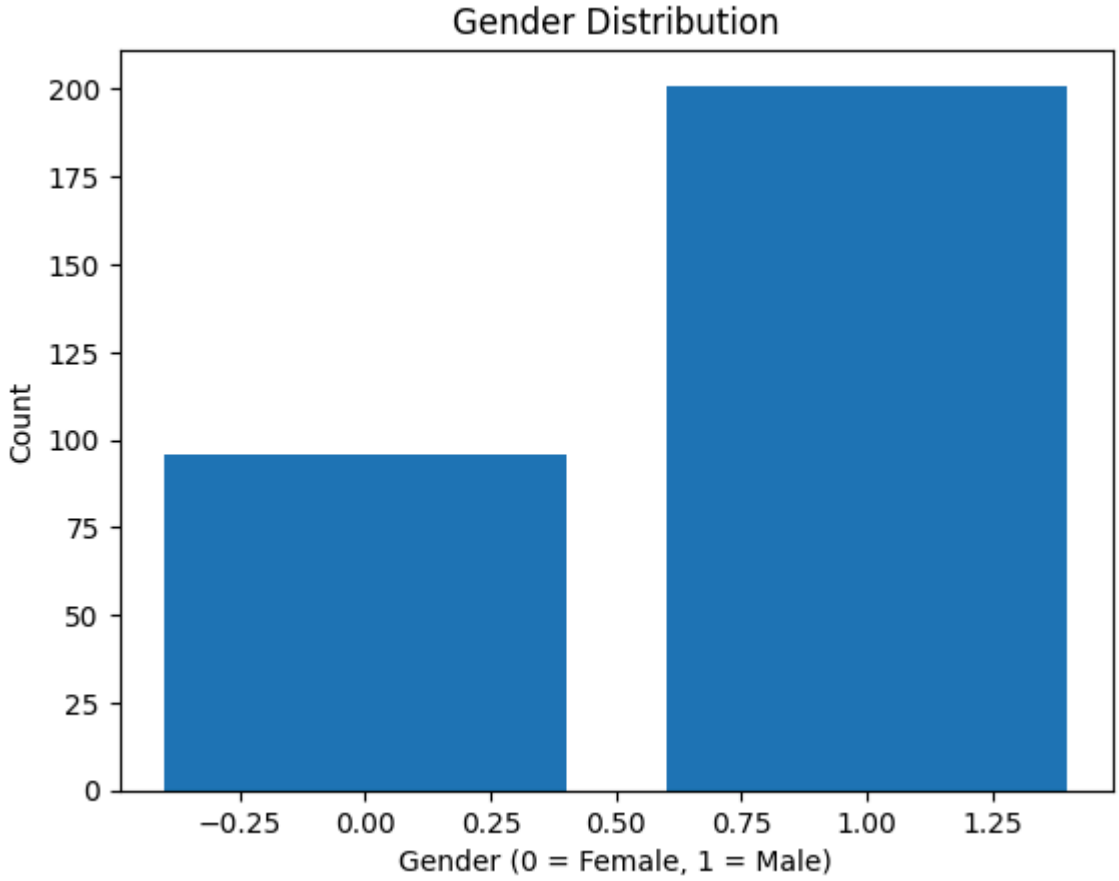
In [5]:
```
# Display Statistics
df.describe()
```

Out[5]:

	age	sex	cp	trestbps	chol	fbs	reste
count	297.000000	297.000000	297.000000	297.000000	297.000000	297.000000	297.000(
mean	54.542088	0.676768	2.158249	131.693603	247.350168	0.144781	0.996(
std	9.049736	0.468500	0.964859	17.762806	51.997583	0.352474	0.994(
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000(
25%	48.000000	0.000000	2.000000	120.000000	211.000000	0.000000	0.000(
50%	56.000000	1.000000	2.000000	130.000000	243.000000	0.000000	1.000(
75%	61.000000	1.000000	3.000000	140.000000	276.000000	0.000000	2.000(
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000(

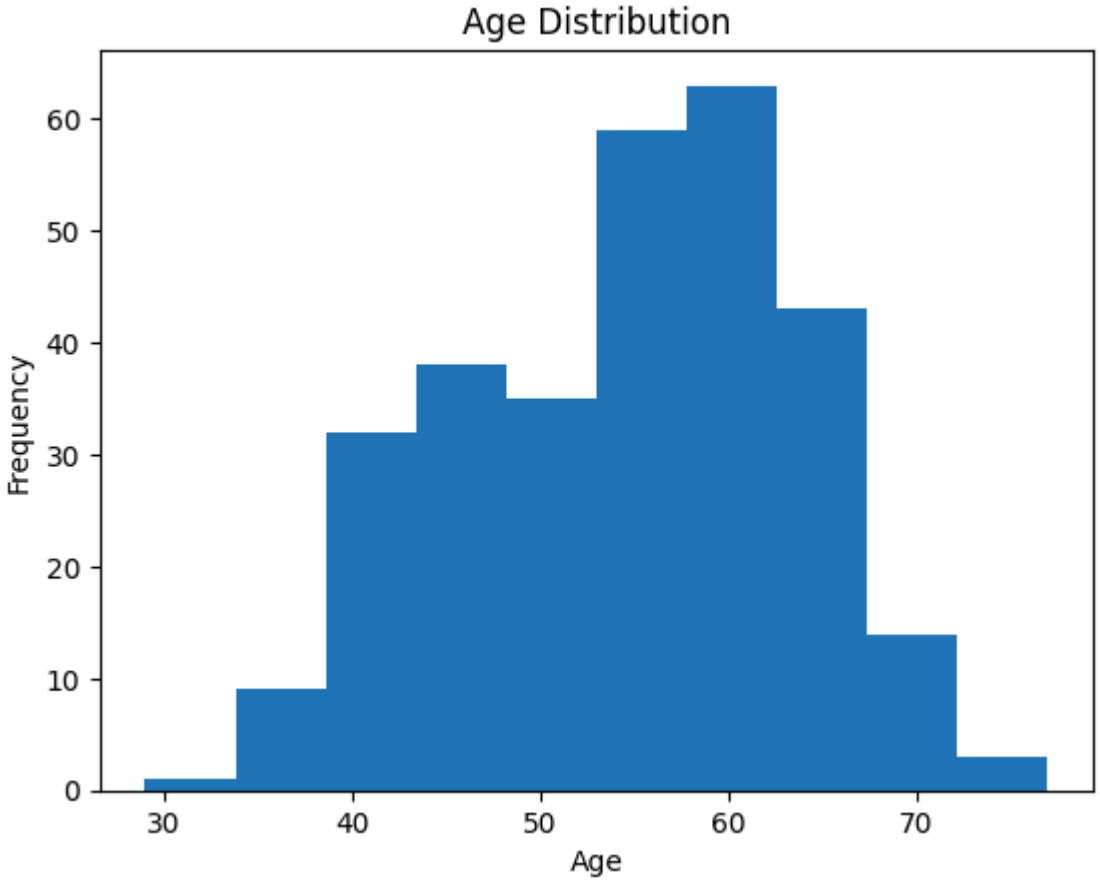
In [51]:
```
# 2.Gender Distribution Analysis

#Count number of males and females
import numpy as np
import matplotlib.pyplot as plt
gender_count = df['sex'].value_counts()
print(gender_count)

#Calculate percentage distribution using NumPy
gender_percentage = (gender_count.values / np.sum(gender_count.values)) * 100
print(gender_percentage)

plt.bar(gender_count.index, gender_count.values)
plt.xlabel("Gender (0 = Female, 1 = Male)")
plt.ylabel("Count")
plt.title("Gender Distribution")
plt.show()
```

```
sex
1    201
0     96
Name: count, dtype: int64
[67.67676768 32.32323232]
```

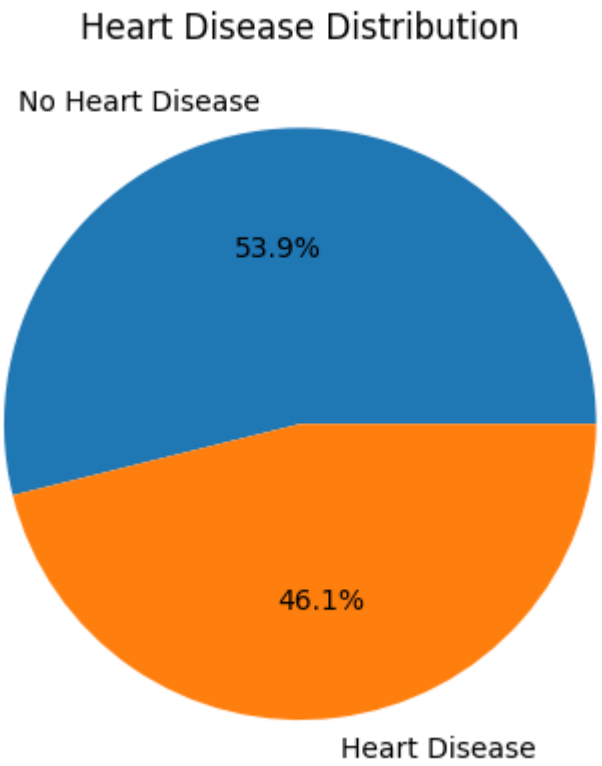


In [52]:
```python
# 3.Age Analysis

# to find  age
print("Minimum Age :", df['age'].min())
print("Maximum Age :", df['age'].max())
print("Mean Age    :", df['age'].mean())
print("Median Age  :", df['age'].median())


import matplotlib.pyplot as plt
plt.hist(df['age'], bins=10)
plt.xlabel("Age")
plt.ylabel("Frequency")
plt.title("Age Distribution")
plt.show()
```

```
Minimum Age : 29
Maximum Age : 77
Mean Age    : 54.54208754208754
Median Age  : 56.0
```



In [20]:
```
# 4.Target Variable Analysis

#Count number of patients with and without heart disease
df['condition'].value_counts()

#Plot pie chart
import matplotlib.pyplot as plt
counts = df['condition'].value_counts()
plt.figure()
plt.pie(counts, labels=['No Heart Disease', 'Heart Disease'], autopct='%1.1f%%')
plt.title('Heart Disease Distribution')
plt.show()

#Calculate disease percentage
(df['condition'].value_counts(normalize=True) * 100)
```

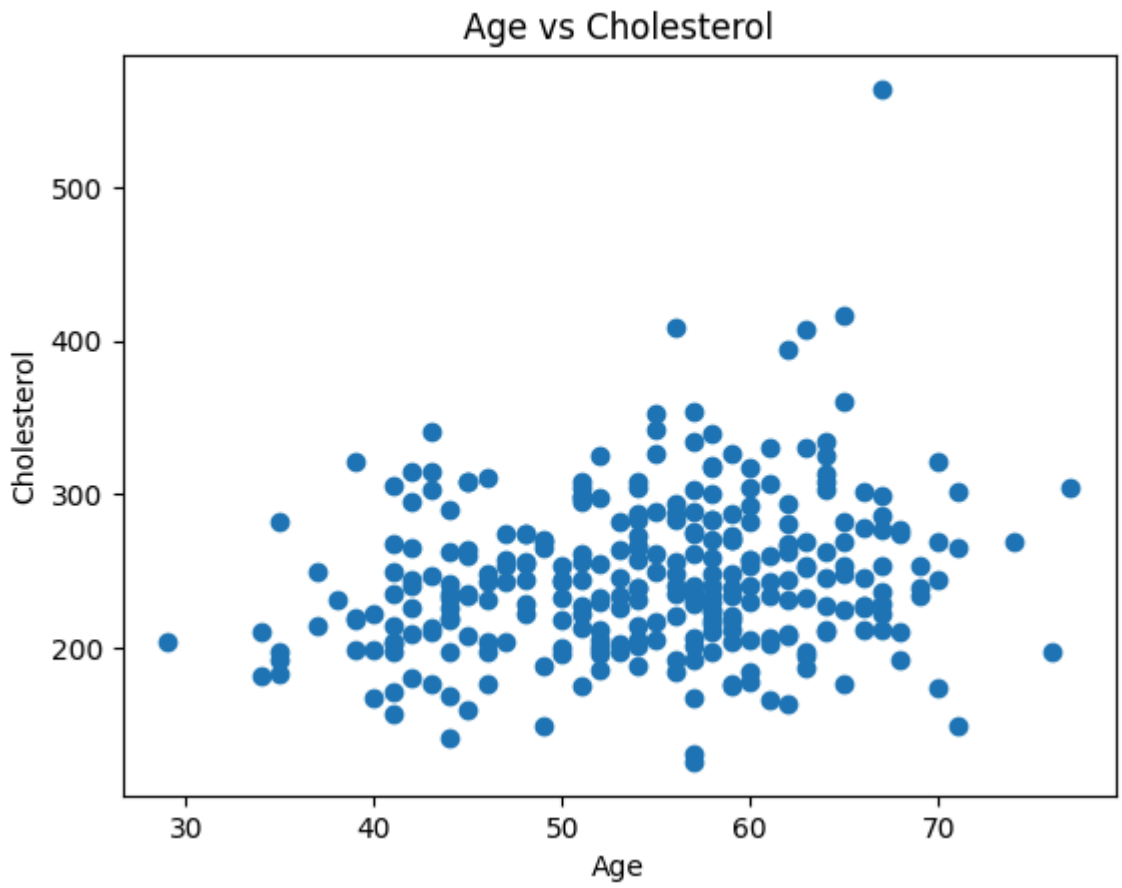


Out[20]:
```
condition
0    53.872054
1    46.127946
Name: proportion, dtype: float64
```

In [34]:
```
# 5.Correlation Between Age and Cholesterol

#Calculate correlation using df.corr()
correlation_matrix = df.corr()
print("Correlation Matrix:\n",correlation_matrix)

#scatter plot (Age vs Cholesterol)
df.corr().loc['age', 'chol']
import matplotlib.pyplot as plt
plt.figure()
plt.scatter(df['age'], df['chol'])
plt.xlabel("Age")
plt.ylabel("Cholesterol")
plt.title("Age vs Cholesterol")
plt.show()
```

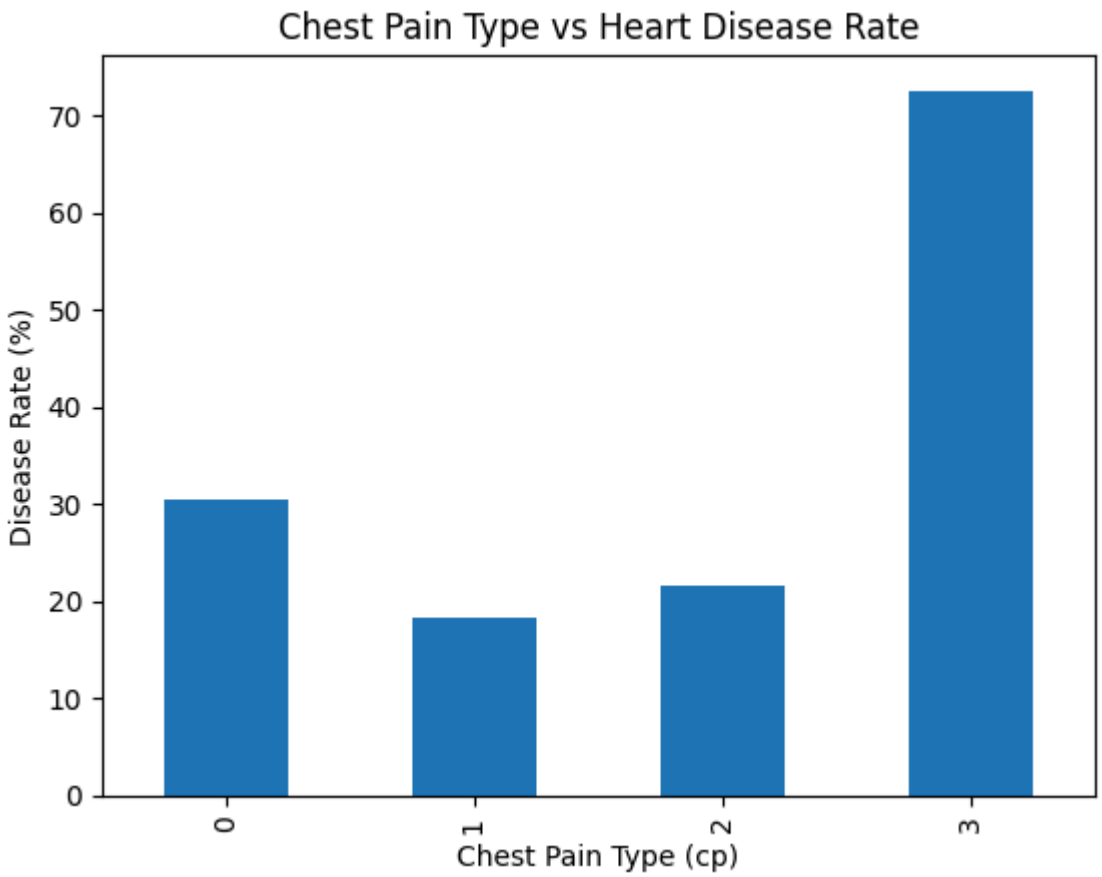
Correlation Matrix:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	condition
age	1.000000	-0.092399	0.110471	0.290476	0.202644	0.132062	0.149917	-0.394563	0.096489	0.197123	0.159405	0.362210	0.120795	0.227075
sex	-0.092399	1.000000	0.008908	-0.066340	-0.198089	0.038850	0.033897	-0.060496	0.143581	0.106567	0.033345	0.091925	0.370556	0.278467
cp	0.110471	0.008908	1.000000	-0.036980	0.072088	-0.057663	0.063905	-0.339308	0.377525	0.203244	0.151079	0.235644	0.266275	0.408945
trestbps	0.290476	-0.066340	-0.036980	1.000000	0.131536	0.180860	0.149242	-0.049108	0.066691	0.191243	0.121172	0.097954	0.130612	0.153490
chol	0.202644	-0.198089	0.072088	0.131536	1.000000	0.012708	0.165046	-0.000075	0.059339	0.038596	-0.009215	0.115945	0.023441	0.080285
fbs	0.132062	0.038850	-0.057663	0.180860	0.012708	1.000000	0.068831	-0.007842	-0.000893	0.008311	0.047819	0.152086	0.051038	0.003167
restecg	0.149917	0.033897	0.063905	0.149242	0.165046	0.068831	1.000000	-0.072290	0.081874	0.113726	0.135141	0.129021	0.013612	0.166343
thalach	-0.394563	-0.060496	-0.339308	-0.049108	-0.000075	-0.007842	-0.072290	1.000000	-0.384368	-0.347640	-0.389307	-0.268727	-0.258386	-0.423817
exang	0.096489	0.143581	0.377525	0.066691	0.059339	-0.000893	0.081874	-0.384368	1.000000	0.289310	0.250572	0.148232	0.323268	0.421355
oldpeak	0.197123	0.106567	0.203244	0.191243	0.038596	0.008311	0.113726	-0.347640	0.289310	1.000000	0.579037	0.294452	0.336809	0.424052
slope	0.159405	0.033345	0.151079	0.121172	-0.009215	0.047819	0.135141	-0.389307	0.250572	0.579037	1.000000	0.109761	0.260096	0.333049
ca	0.362210	0.091925	0.235644	0.097954	0.115945	0.152086	0.129021	-0.268727	0.148232	0.294452	0.109761	1.000000	0.248825	0.463189
thal	0.120795	0.370556	0.266275	0.130612	0.023441	0.051038	0.013612	-0.258386	0.323268	0.336809	0.260096	0.248825	1.000000	0.520516
condition	0.227075	0.278467	0.408945	0.153490	0.080285	0.003167	0.166343	-0.423817	0.421355	0.424052	0.333049	0.463189	0.520516	1.000000



In [35]:
```python
# 6.Chest Pain Type vs Disease

#Group by cp and calculate disease rate
cp_disease_rate = df.groupby('cp')['condition'].mean() * 100
print("Disease Rate (%) by Chest Pain Type:\n")
print(cp_disease_rate)

#bar chart
cp_disease_rate.plot(kind='bar')
plt.xlabel("Chest Pain Type (cp)")
plt.ylabel("Disease Rate (%)")
plt.title("Chest Pain Type vs Heart Disease Rate")
plt.show()

#Identify which chest pain type is most risky
most_risky_cp = cp_disease_rate.idxmax()
print("\nMost risky chest pain type:", most_risky_cp)
```

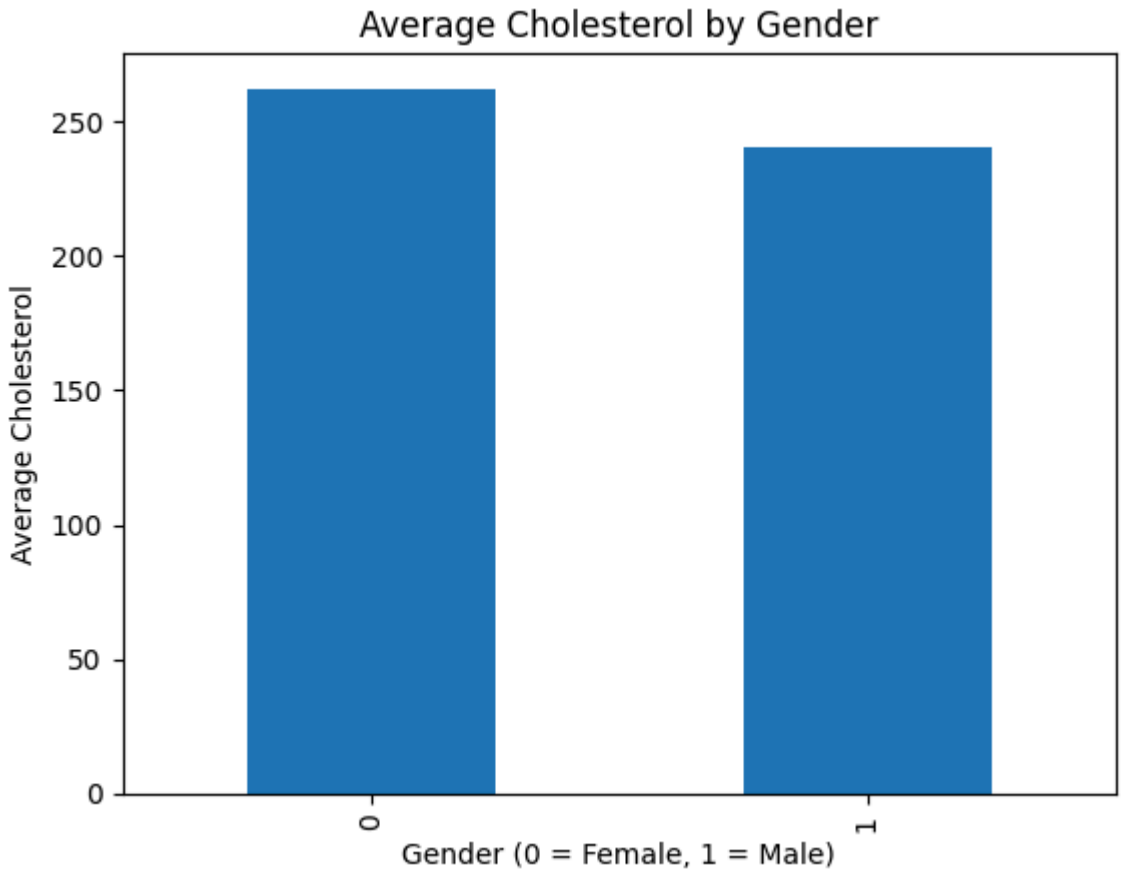
```
Disease Rate (%) by Chest Pain Type:

cp
0    30.434783
1    18.367347
2    21.686747
3    72.535211
Name: condition, dtype: float64
```




```
Most risky chest pain type: 3
```

In [36]:
```
# 7.Average Cholesterol by Gender

#Group by sex
avg_chol_gender = df.groupby('sex')['chol'].mean()

#Calculate mean cholesterol
print("Average Cholesterol by Gender:\n")
print(avg_chol_gender)

#bar plot
avg_chol_gender.plot(kind='bar')
plt.xlabel("Gender (0 = Female, 1 = Male)")
plt.ylabel("Average Cholesterol")
plt.title("Average Cholesterol by Gender")
plt.show()
```

```
Average Cholesterol by Gender:

sex
0    262.229167
1    240.243781
Name: chol, dtype: float64
```



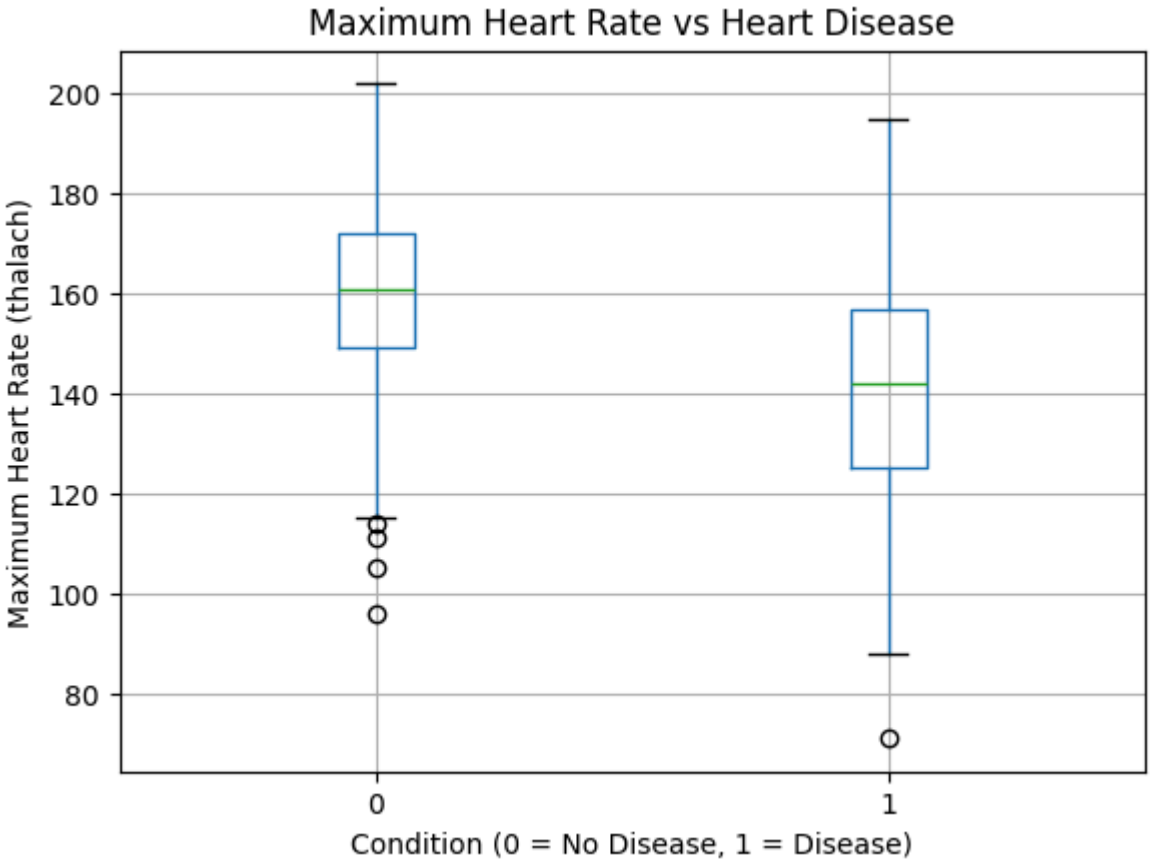
In [38]:
```
# 8.Resting Blood Pressure Analysis

# Average BP
avg_bp = df['trestbps'].mean()
print("Average Resting Blood Pressure:", avg_bp)

# Patients with BP > 140
high_bp = df[df['trestbps'] > 140]
print("Number of patients with BP > 140:", high_bp.shape[0])

# Compare disease presence in high BP group
high_bp_disease_rate = high_bp['condition'].value_counts(normalize=True) * 100
print("\nDisease presence in high BP group (%):")
print(high_bp_disease_rate)
```

```
Average Resting Blood Pressure: 131.69360269360268
Number of patients with BP > 140: 66

Disease presence in high BP group (%):
condition
1    59.090909
0    40.909091
Name: proportion, dtype: float64
```

In [39]:
```
# 9.Maximum Heart Rate vs Disease

# Average maximum heart rate for disease and non-disease patients
avg_thalach = df.groupby('condition')['thalach'].mean()

print("Average Maximum Heart Rate (thalach):\n")
print("No Heart Disease (0):", avg_thalach[0])
print("Heart Disease (1):", avg_thalach[1])
```

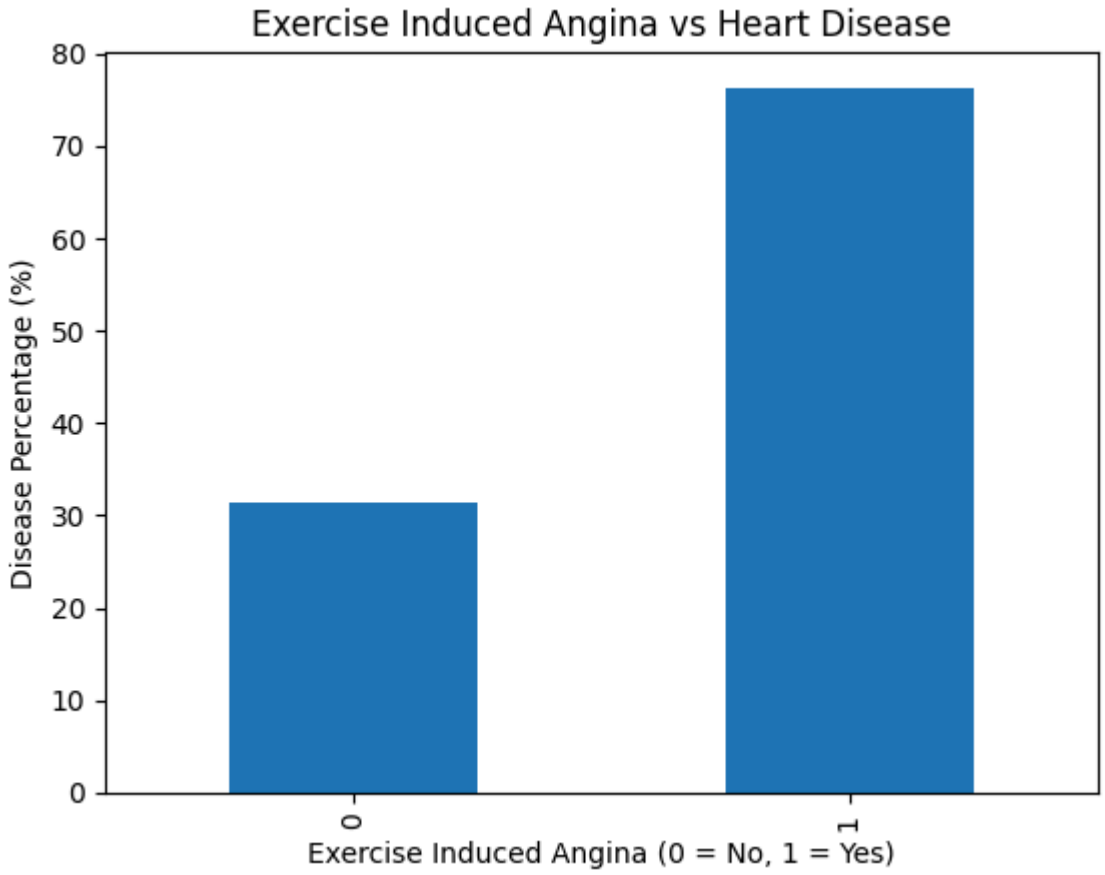
```
# Boxplot
df.boxplot(column='thalach', by='condition')
plt.xlabel("Condition (0 = No Disease, 1 = Disease)")
plt.ylabel("Maximum Heart Rate (thalach)")
plt.title("Maximum Heart Rate vs Heart Disease")
plt.suptitle("")  # remove automatic subtitle
plt.show()
```

```
Average Maximum Heart Rate (thalach):

No Heart Disease (0): 158.58125
Heart Disease (1): 139.1094890510949
```



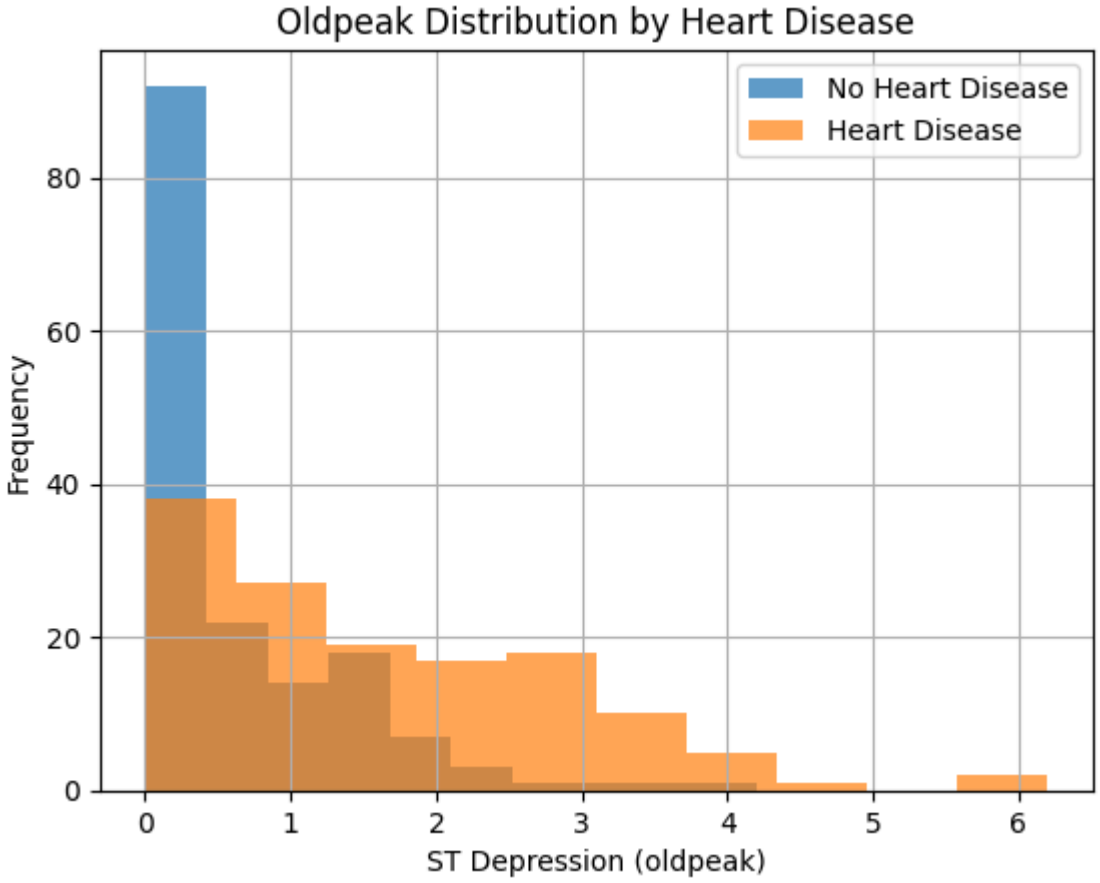
In [41]:
```
# 10.Exercise Induced Angina Impact

#Calculate disease percentage in:
#exang = 0
#exang = 1

exang_disease_rate = df.groupby('exang')['condition'].mean() * 100
print("Disease Percentage by Exercise Induced Angina:\n")
print("exang = 0 (No Angina):", exang_disease_rate[0], "%")
print("exang = 1 (Angina):", exang_disease_rate[1], "%")

#bar chart
exang_disease_rate.plot(kind='bar')
plt.xlabel("Exercise Induced Angina (0 = No, 1 = Yes)")
plt.ylabel("Disease Percentage (%)")
plt.title("Exercise Induced Angina vs Heart Disease")
plt.show()
```

```
Disease Percentage by Exercise Induced Angina:

exang = 0 (No Angina): 31.5 %
exang = 1 (Angina): 76.28865979381443 %
```



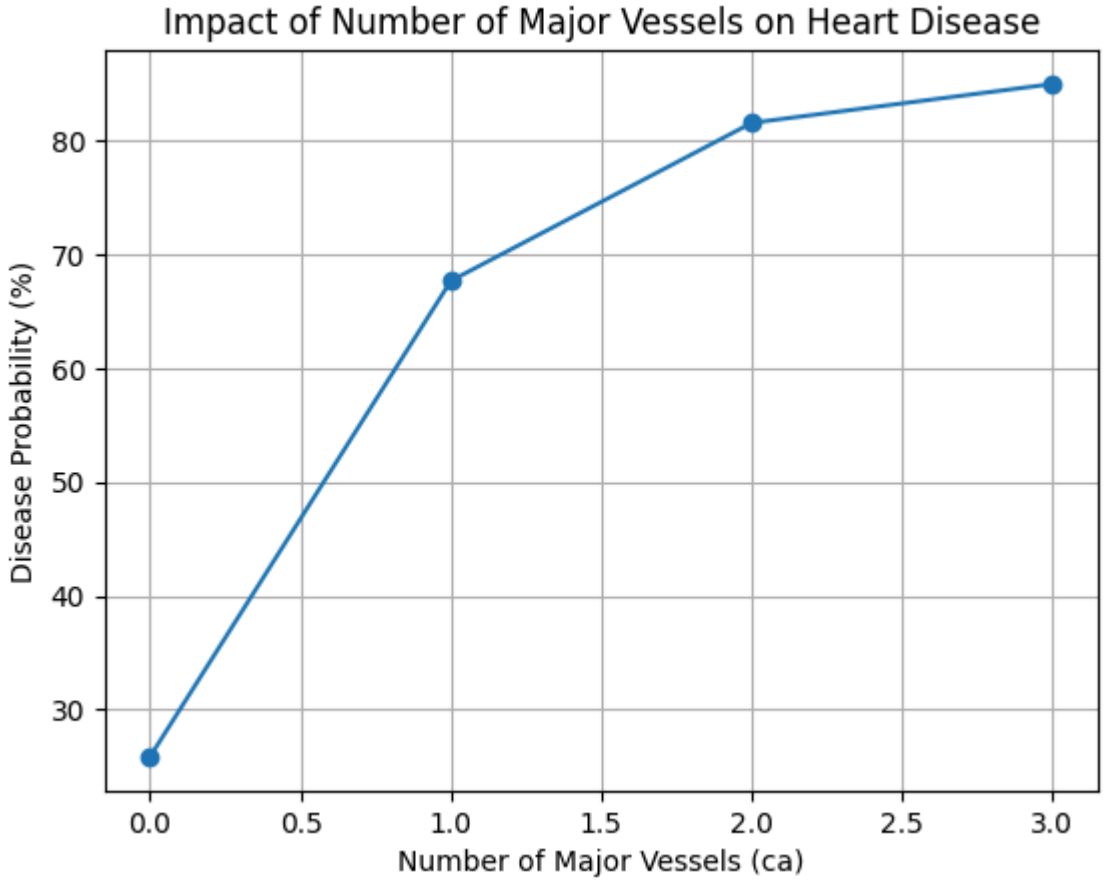
In [42]:
```python
# 11.ST Depression (oldpeak) Analysis

#Calculate mean oldpeak by target
mean_oldpeak = df.groupby('condition')['oldpeak'].mean()
print("Mean ST Depression (oldpeak):\n")
print("No Heart Disease (0):", mean_oldpeak[0])
print("Heart Disease (1):", mean_oldpeak[1])

#histogram
df[df['condition'] == 0]['oldpeak'].hist(alpha=0.7)
df[df['condition'] == 1]['oldpeak'].hist(alpha=0.7)
plt.xlabel("ST Depression (oldpeak)")
plt.ylabel("Frequency")
plt.title("Oldpeak Distribution by Heart Disease")
plt.legend(["No Heart Disease", "Heart Disease"])
plt.show()
```

```
Mean ST Depression (oldpeak):

No Heart Disease (0): 0.59875
Heart Disease (1): 1.5890510948905108
```



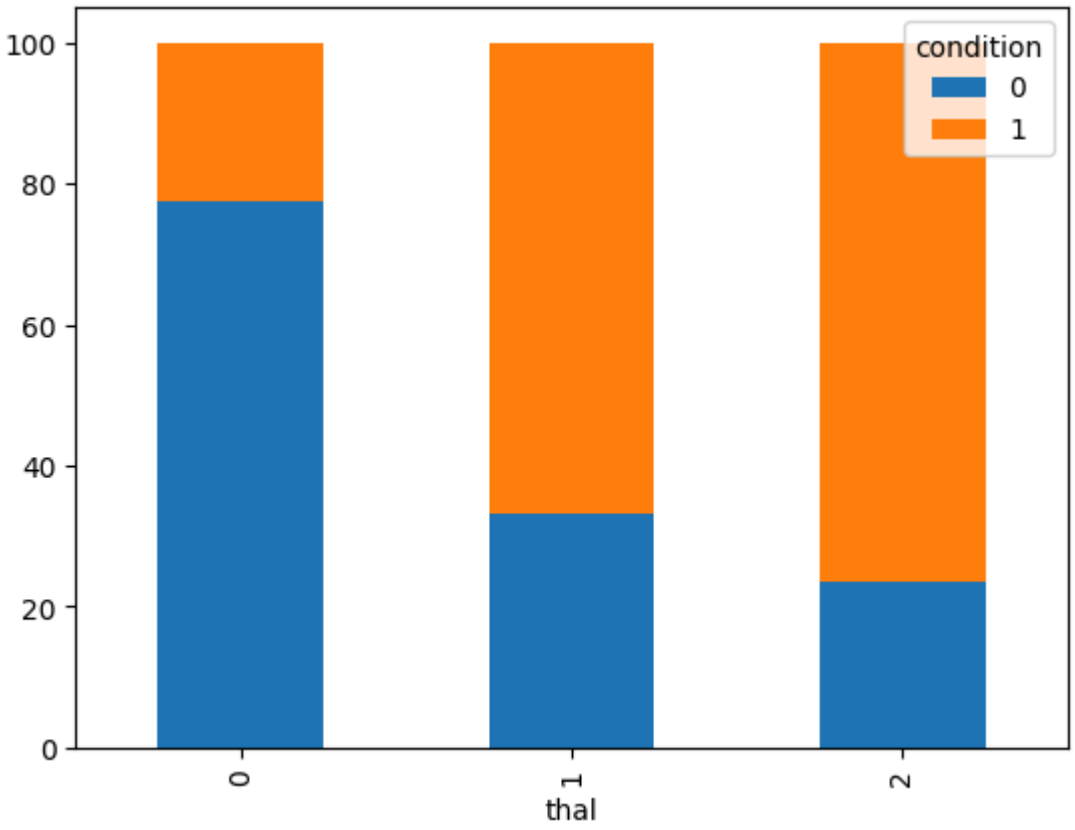
In [45]:
```
# 12.Number of Major Vessels (ca) Impact

#Group by ca Calculate disease probability
ca_disease_prob = df.groupby('ca')['condition'].mean() * 100
print("Disease Probability (%) by Number of Major Vessels (ca):\n")
print(ca_disease_prob)

#Plot line chart
ca_disease_prob.plot(marker='o')
plt.xlabel("Number of Major Vessels (ca)")
plt.ylabel("Disease Probability (%)")
plt.title("Impact of Number of Major Vessels on Heart Disease")
plt.grid(True)
plt.show()
```

```
Disease Probability (%) by Number of Major Vessels (ca):

ca
0    25.862069
1    67.692308
2    81.578947
3    85.000000
Name: condition, dtype: float64
```



In [47]:
```
# 13.Thalassemia vs Disease

#Cross-tabulate thal and target
thal_ct = pd.crosstab(df['thal'], df['condition'])

#Convert to percentage
thal_percent = thal_ct.div(thal_ct.sum(axis=1), axis=0) * 100

print("Thalassemia vs Heart Disease (Percentage):\n")
print(thal_percent)

#Plot stacked bar chart
thal_percent.plot(kind='bar', stacked=True)
plt.show()
```

```
Thalassemia vs Heart Disease (Percentage):

condition          0          1
thal
0          77.439024  22.560976
1          33.333333  66.666667
2          23.478261  76.521739
```

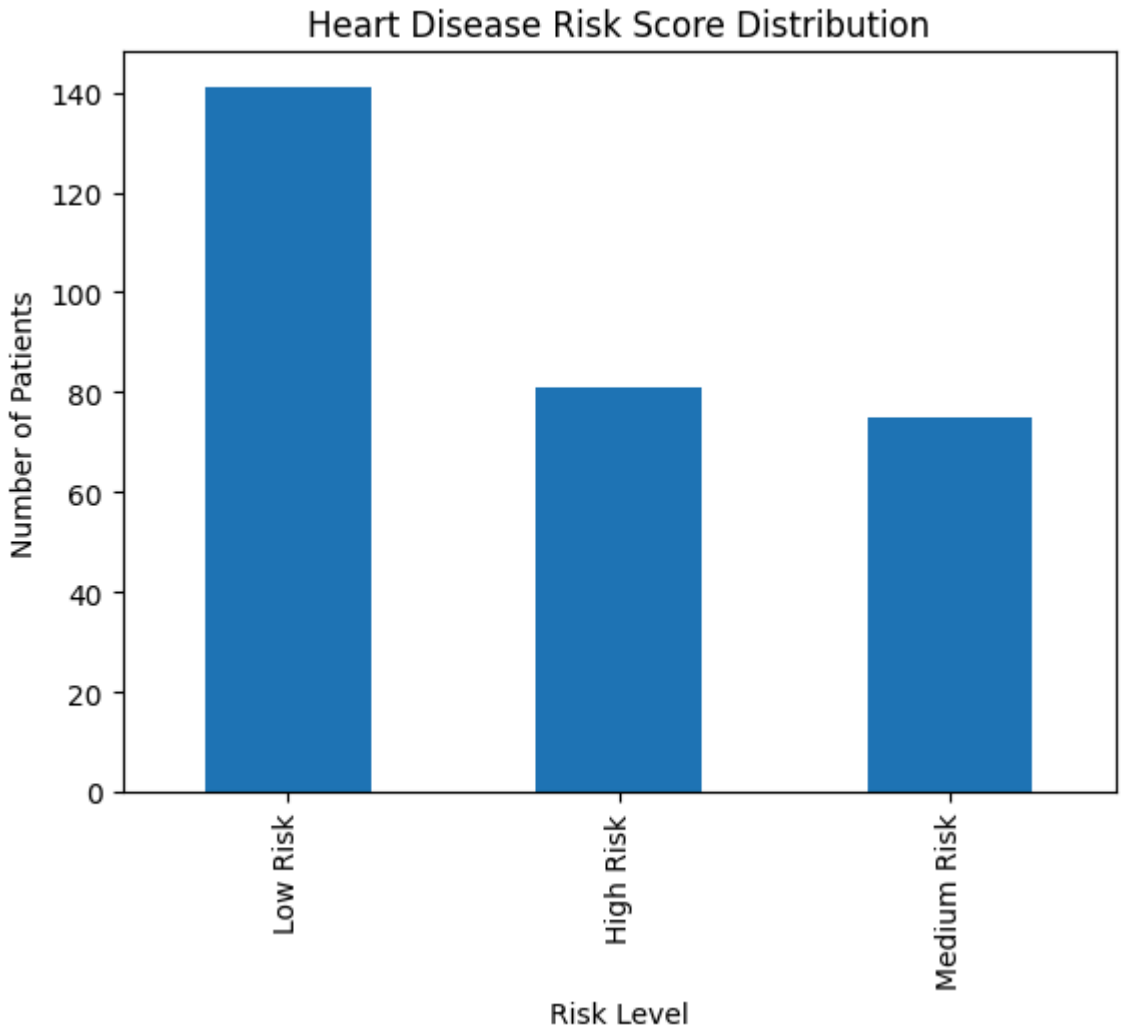


In [49]:
```
# 14.Multi-Factor Risk Analysis

high_risk = df[
    (df['age'] > 50) &
    (df['chol'] > 240) &
    (df['trestbps'] > 140)
]

disease_percentage = high_risk['condition'].mean() * 100
print("Number of high-risk patients:", high_risk.shape[0])
print("Percentage having heart disease:", disease_percentage)
```

```
Number of high-risk patients: 33
Percentage having heart disease: 66.66666666666666
```

In [50]:
```
# 15.Create Risk Score (Custom Analysis)

df['risk_score'] = (df['chol'] / 200) + (df['trestbps'] / 120) + df['oldpeak']
df['risk_level'] = 'Low Risk'
df.loc[df['risk_score'] >= 3, 'risk_level'] = 'Medium Risk'
df.loc[df['risk_score'] >= 4, 'risk_level'] = 'High Risk'

# Display risk level counts
print("Risk Level Distribution:\n")
print(df['risk_level'].value_counts())

# Visualize distribution
df['risk_level'].value_counts().plot(kind='bar')
plt.xlabel("Risk Level")
plt.ylabel("Number of Patients")
plt.title("Heart Disease Risk Score Distribution")
plt.show()
```

```
Risk Level Distribution:

risk_level
Low Risk       141
High Risk       81
Medium Risk     75
Name: count, dtype: int64
```



In [ ]:

```
1.Does cholesterol strongly impact heart disease?

No not strongly
Cholesterol shows only a weak to moderate association with heart disease
Many patients with normal cholesterol still have heart disease and vice versa
Cholesterol alone is not a strong predictor without other risk factors
Conclusion: Cholesterol contributes to risk but does not strongly impact heart dis

2.Is the male population more vulnerable?

Yes
Male patients show a higher prevalence of heart disease compared to females
Gender-based analysis indicates males are more prone to heart disease
Conclusion:The male population is more vulnerable to heart disease

3.Does exercise-induced angina significantly increase risk?

Yes significantly
Patients with exercise-induced angina (exang = 1) have a much higher disease perce
This feature shows a clear and strong separation between diseased and non-diseased
Conclusion:Exercise-induced angina is a strong indicator of heart disease risk
```

```
4.Which feature has the strongest correlation with disease

Number of major vessels (ca)
ca shows the strongest positive correlation with heart disease
Disease probability increases steadily as the number of major vessels increases
Other strong features include
Chest pain type (cp)
ST depression (oldpeak)
Exercise-induced angina (exang)
```