Part 1 Diamonds

```
# Task1
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns


import warnings
warnings.filterwarnings('ignore')


# Task2
df = pd.read_csv('diamond.csv')
print("Dataset loaded successfully.")
print("\n-------DATASET OVERVIEW-------")
print(df.info())
print("\n-------DATASET DESCRIPTION-------")
print(df.describe(include='all'))
print("\n-------MISSING VALUES-------")
print(df.isnull().sum())
```

```
Dataset loaded successfully.

-------DATASET OVERVIEW-------
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53940 entries, 0 to 53939
Data columns (total 13 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   Unnamed: 0                          53563 non-null  object
 1   carat                               52430 non-null  object
 2   cut                                 52647 non-null  object
 3   color                               52428 non-null  object
 4   clarity                             53587 non-null  object
 5   average us salary                   53940 non-null  int64
 6   number of diamonds mined (millions)  53940 non-null  float64
 7   depth                               53246 non-null  object
 8   table                               52398 non-null  object
 9   price                               52357 non-null  object
 10  x                                   52414 non-null  object
 11  y                                   52719 non-null  object
 12  z                                   52507 non-null  object
dtypes: float64(1), int64(1), object(11)
memory usage: 5.4+ MB
None

-------DATASET DESCRIPTION-------
       Unnamed: 0  carat    cut  color clarity  average us salary  \
count       53563  52430  52647  52428   53587       53940.000000
unique      52181    276     18     17      18                NaN
top         FALSE    0.3  Ideal      G     SI1                NaN
freq          480   2469  19938  10588   12592                NaN
mean          NaN    NaN    NaN    NaN     NaN       39521.990100
std           NaN    NaN    NaN    NaN     NaN        5486.892971
min           NaN    NaN    NaN    NaN     NaN       30000.000000
25%           NaN    NaN    NaN    NaN     NaN       34780.000000
50%           NaN    NaN    NaN    NaN     NaN       39547.500000
75%           NaN    NaN    NaN    NaN     NaN       44252.000000
max           NaN    NaN    NaN    NaN     NaN       48999.000000

        number of diamonds mined (millions)  depth  table  price      x  \
count                          53940.000000  53246  52398  52357  52414
unique                                  NaN    187    127  11443    556
top                                     NaN     62     56  MAYBE  FALSE
freq                                    NaN   2163   9377    180    488
mean                               2.902669    NaN    NaN    NaN    NaN
std                                1.325985    NaN    NaN    NaN    NaN
min                                0.600000    NaN    NaN    NaN    NaN
25%                                1.750000    NaN    NaN    NaN    NaN
50%                                2.910000    NaN    NaN    NaN    NaN
75%                                4.050000    NaN    NaN    NaN    NaN
max                                5.200000    NaN    NaN    NaN    NaN

            y      z
count   52719  52507
unique    552    378
top      4.34    2.7
freq      422    735
mean      NaN    NaN
```

```
df.rename(columns={"Unnamed: 0": "sno"}, inplace=True)


df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53940 entries, 0 to 53939
Data columns (total 13 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   sno                                 53563 non-null  object
 1   carat                               52430 non-null  object
 2   cut                                 52647 non-null  object
 3   color                               52428 non-null  object
 4   clarity                             53587 non-null  object
 5   average us salary                   53940 non-null  int64
 6   number of diamonds mined (millions)  53940 non-null  float64
 7   depth                               53246 non-null  object
 8   table                               52398 non-null  object
 9   price                               52357 non-null  object
 10  x                                   52414 non-null  object
 11  y                                   52719 non-null  object
 12  z                                   52507 non-null  object
dtypes: float64(1), int64(1), object(11)
memory usage: 5.4+ MB
```

```
# Task 3
numeric_columns = ['carat', 'depth', 'table', 'price', 'x', 'y', 'z']
for column in numeric_columns:
    df[column] = pd.to_numeric(df[column], errors='coerce')


for column in numeric_columns:
    median_value = df[column].median()
    df[column].fillna(median_value, inplace=True)


unexpected_entries = ["true", "false", "maybe"]
for column in df.columns:
    df[column] = df[column].apply(lambda x: np.nan if str(x).lower() in unexpected_entries else x)

df.dropna(inplace=True)


# handling unexpected entries
unique_values_summary = {}
for column in df.columns:
    unique_values = df[column].unique()
    unique_values_summary[column] = unique_values
    print(f"Column '{column}' has {len(unique_values)} unique values:")
    print(unique_values)
    print("\n" + "-"*50 + "\n")
```

```
Column 'sno' has 49266 unique values:
['1' '2' '3' ... '53938' '53939' '53940']

--------------------------------------------------

Column 'carat' has 273 unique values:
[0.23 0.21 0.29 0.31 0.24 0.22 0.3  0.2  0.32 0.26 0.33 0.25 0.35 0.42
 0.28 0.38 0.7  0.86 0.71 0.78 0.8  0.75 0.74 0.81 0.59 0.9  0.73 0.91
 0.61 0.77 0.76 0.64 0.72 0.79 0.58 1.17 0.6  0.83 0.54 0.98 0.52 0.53
 0.51 1.01 1.05 0.55 0.63 0.87 1.   0.57 0.82 0.96 1.04 0.93 1.2  0.99
 0.34 0.43 0.36 0.84 0.89 1.02 0.56 0.85 0.92 0.95 1.27 0.66 1.12 0.68
 1.03 0.97 0.62 1.22 1.08 0.88 0.5  1.19 0.39 0.65 1.24 1.5  0.27 0.41
 1.13 1.06 0.69 0.4  1.14 0.94 1.29 1.52 1.16 1.21 1.23 1.09 0.67 1.11
 1.1  1.15 1.25 1.18 1.07 1.28 1.51 0.37 1.31 1.26 1.39 1.44 1.35 1.3
 1.41 1.36 1.32 1.45 1.58 1.54 1.34 1.38 1.33 1.74 1.47 1.95 2.   1.37
 1.83 1.62 1.57 1.69 2.06 1.72 1.66 2.14 1.49 1.46 2.15 1.96 2.22 1.7
 1.64 1.53 1.85 2.01 1.4  2.27 1.56 1.81 1.68 1.55 1.82 2.03 1.73 1.59
 1.42 1.43 2.08 1.48 1.6  2.49 2.02 2.07 2.21 2.1  1.91 2.25 1.65 2.17
 1.71 2.32 2.72 1.61 2.23 2.11 2.05 1.63 2.3  2.31 1.75 2.04 2.12 1.77
 2.5  1.67 1.84 2.2  3.01 3.   1.88 2.33 1.8  2.34 1.9  2.74 1.78 2.16
 1.76 1.79 1.94 2.68 2.43 3.11 2.09 1.89 2.52 1.87 2.19 1.86 2.77 2.63
 3.05 2.46 3.02 2.38 2.24 2.26 2.36 1.99 2.29 3.65 2.45 2.4  2.54 2.18
 3.24 2.13 3.22 3.5  2.28 2.48 1.98 2.44 2.75 1.93 2.41 2.35 2.51 2.7
 2.55 1.97 2.53 2.58 2.37 2.47 2.8  4.01 2.56 3.04 1.92 2.39 3.4  4.
 3.67 2.42 2.66 2.65 2.59 2.6  2.57 2.71 2.61 4.13 2.64 5.01 4.5  2.67
 3.51 0.44 0.45 0.47 0.46 0.48 0.49]

--------------------------------------------------

Column 'cut' has 15 unique values:
['Ideal' 'Premium' 'Good' 'Very Good' 'Fair' 'IDEAL' 'FAIR' 'very good'
```

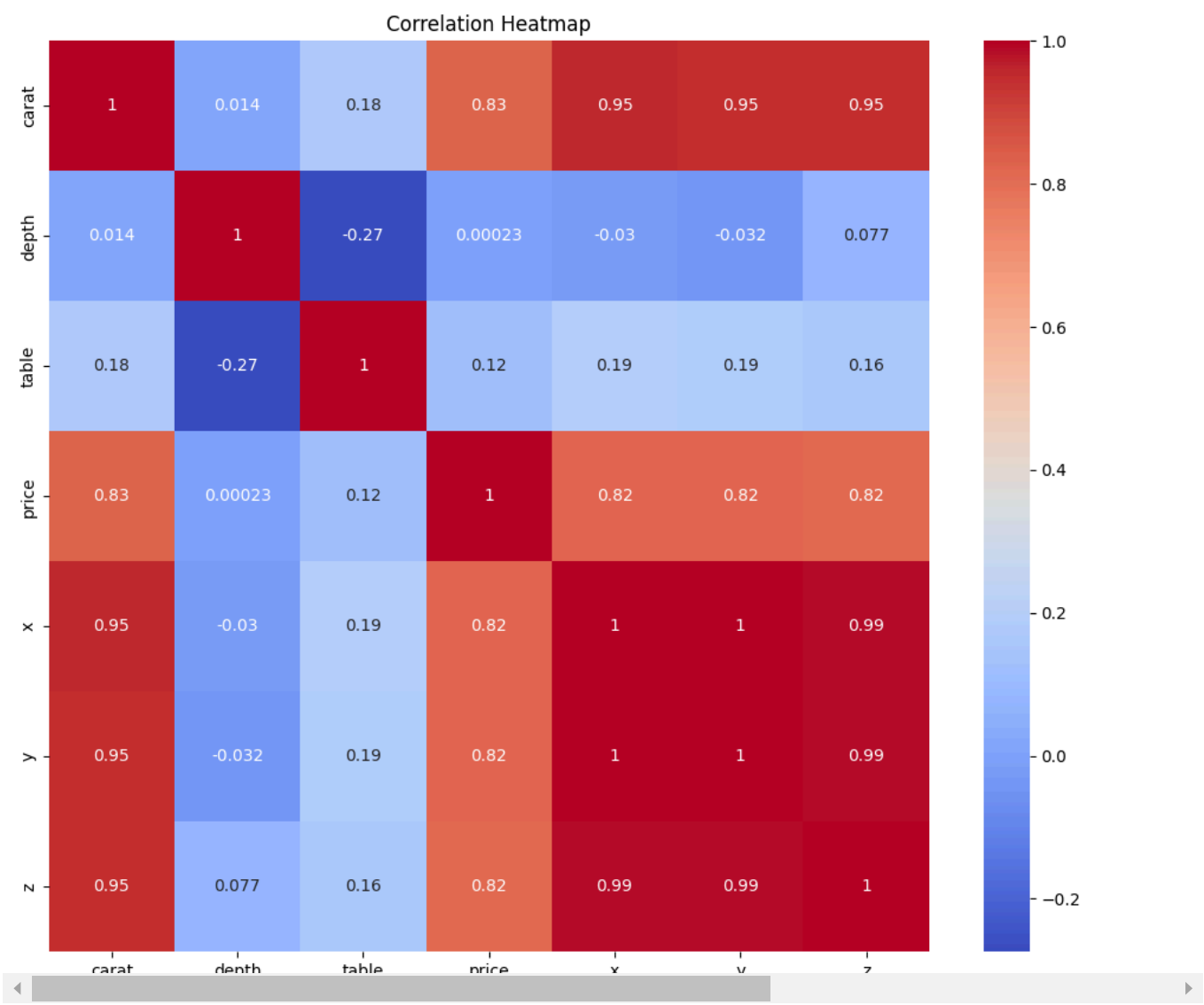
```
 'premium' 'VERY GOOD' 'PREMIUM' 'GOOD' 'ideal' 'good' 'fair']

--------------------------------------------------

Column 'color' has 14 unique values:
['E' 'I' 'J' 'H' 'F' 'G' 'D' 'f' 'g' 'd' 'h' 'j' 'e' 'i']

--------------------------------------------------

Column 'clarity' has 14 unique values:
['SI2' 'SI1' 'VS1' 'VS2' 'VVS2' 'VVS1' 'I1' 'IF' 'si2' 'if' 'vvs2' 'si1'
 'vs2' 'vs1']

--------------------------------------------------

Column 'average us salary' has 17501 unique values:
[31282 40049 33517 ... 40291 41324 46675]

--------------------------------------------------

Column 'number of diamonds mined (millions)' has 461 unique values:
[5.01 1.69 3.85 3.49 4.7  0.86 1.68 4.02 1.2  2.63 2.39 3.95 3.78 5.11
 3.58 2.03 2.48 1.05 4.96 2.86 1.27 4.92 4.29 4.1  1.88 2.29 2.65 3.9
 4.46 5.17 0.66 0.74 1.75 1.3  5.15 0.91 1.48 3.66 1.16 3.96 2.41 4.89
 4.39 2.19 0.62 1.73 2.94 3.17 5.19 3.99 4.38 3.02 2.49 1.79 4.54 2.27
 0.84 5.2  1.99 4.45 0.68 1.37 4.88 3.7  1.9  2.72 3.98 1.61 4.74 1.11
 0.73 4.83 2.14 2.11 2.2  2.68 4.21 3.83 3.19 1.81 4.08 4.4  1.5  2.54
```

```
# Task4
categorical_columns = df.select_dtypes(include=['object']).columns
for column in categorical_columns:
    df[column] = df[column].str.lower()


# Task 5
z_threshold = 3
for column in numeric_columns:

    col_mean = df[column].mean()
    col_std = df[column].std()

    z_scores = (df[column] - col_mean) / col_std

    df[column] = np.where(np.abs(z_scores) > z_threshold, df[column].median(), df[column])



df.info()
print("\n-------MISSING VALUES-------")
print(df.isnull().sum())
```

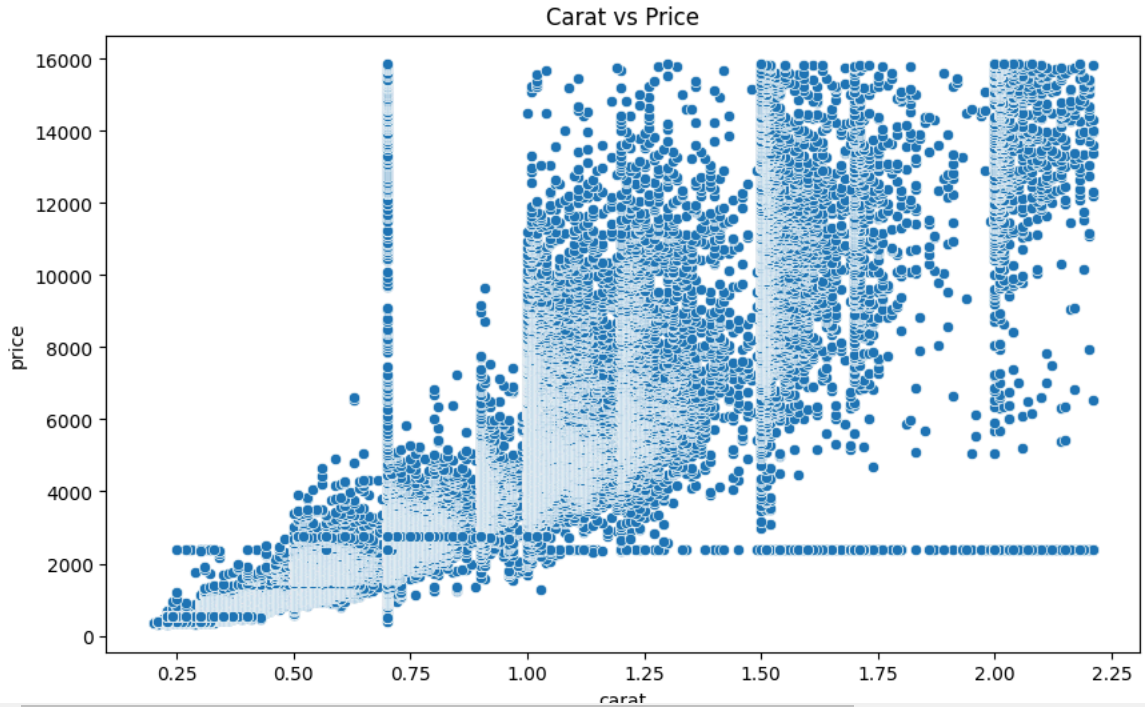
```
<class 'pandas.core.frame.DataFrame'>
Index: 49266 entries, 0 to 53939
Data columns (total 13 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   sno                                 49266 non-null  object
 1   carat                               49266 non-null  float64
 2   cut                                 49266 non-null  object
 3   color                               49266 non-null  object
 4   clarity                             49266 non-null  object
 5   average us salary                   49266 non-null  int64
 6   number of diamonds mined (millions)  49266 non-null  float64
 7   depth                               49266 non-null  float64
 8   table                               49266 non-null  float64
 9   price                               49266 non-null  float64
 10  x                                   49266 non-null  float64
 11  y                                   49266 non-null  float64
 12  z                                   49266 non-null  float64
dtypes: float64(8), int64(1), object(4)
memory usage: 5.3+ MB

-------MISSING VALUES-------
sno                                 0
carat                               0
cut                                 0
color                               0
clarity                             0
average us salary                   0
number of diamonds mined (millions)  0
depth                               0
table                               0
price                               0
x                                   0
y                                   0
z                                   0
```

```
dtype: int64
```

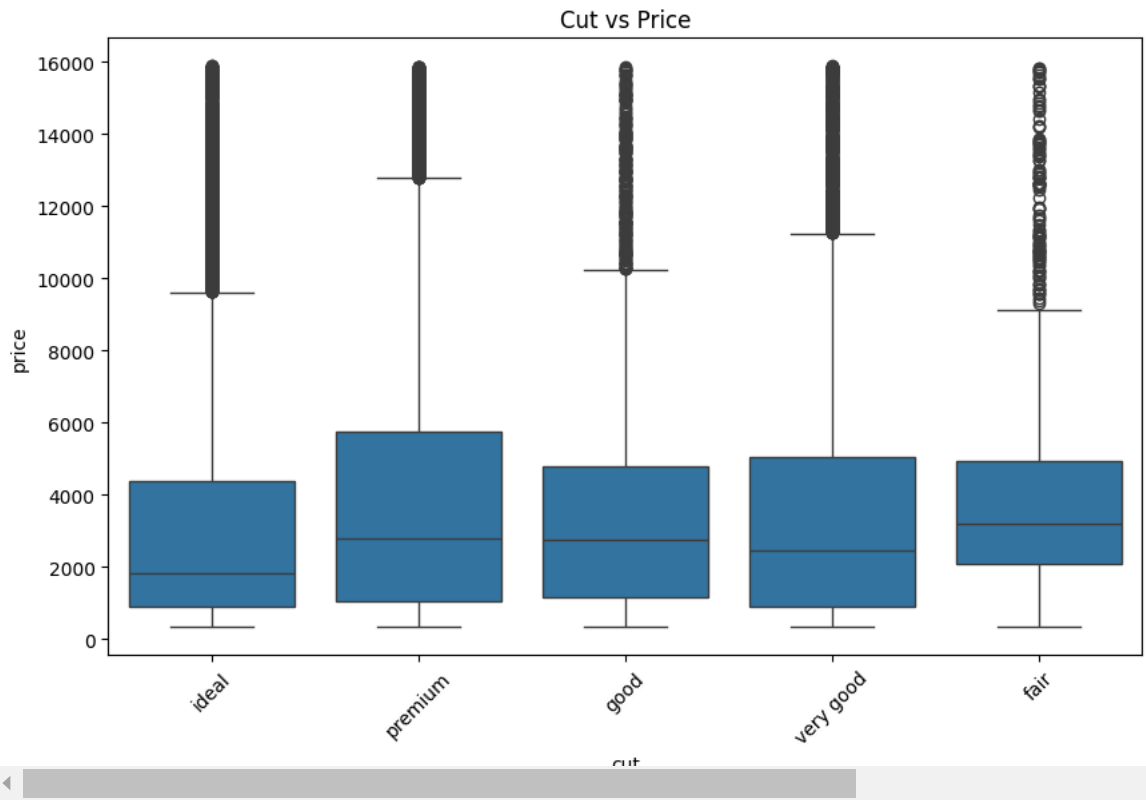
```
# Task6
# Graph 1 - Correlation Heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(df[numeric_columns].corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



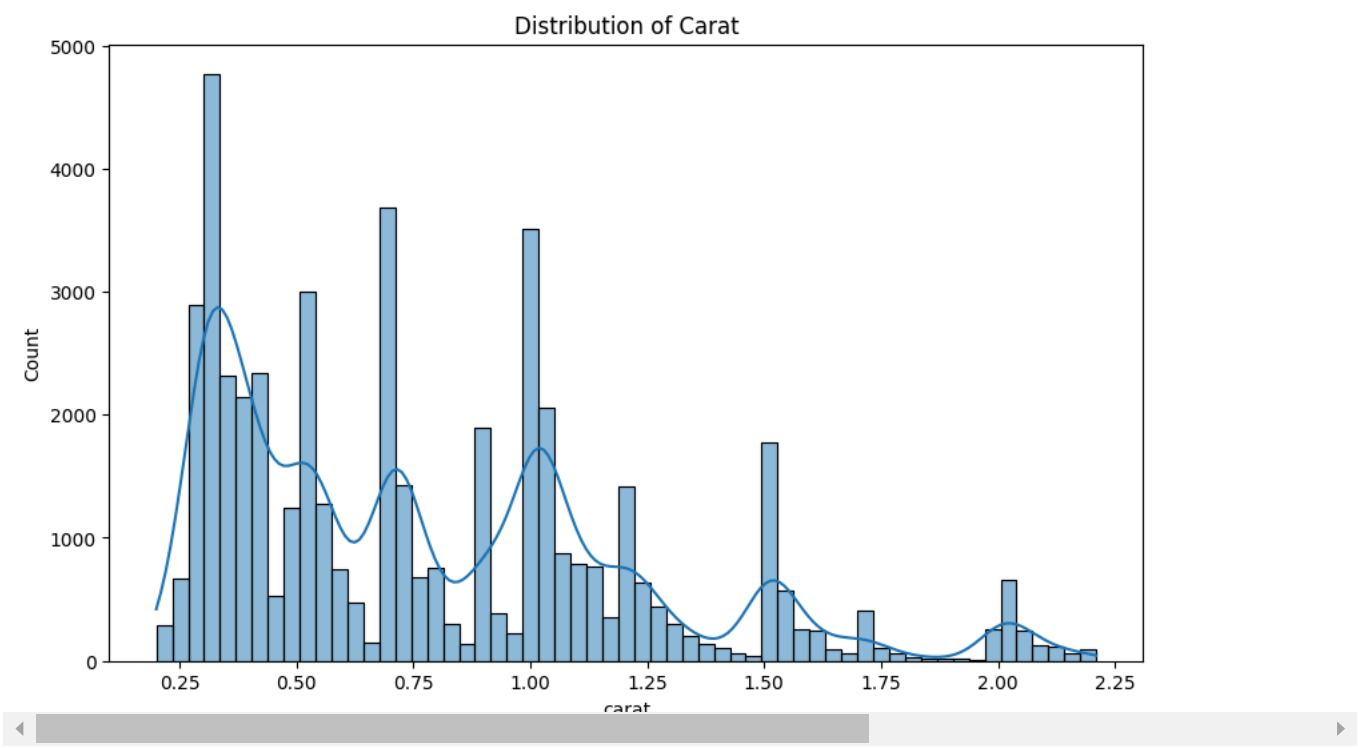

```
# Graph 2 - Scatter plot of carat vs price
plt.figure(figsize=(10, 6))
sns.scatterplot(x='carat', y='price', data=df)
plt.title('Carat vs Price')
plt.show()
```



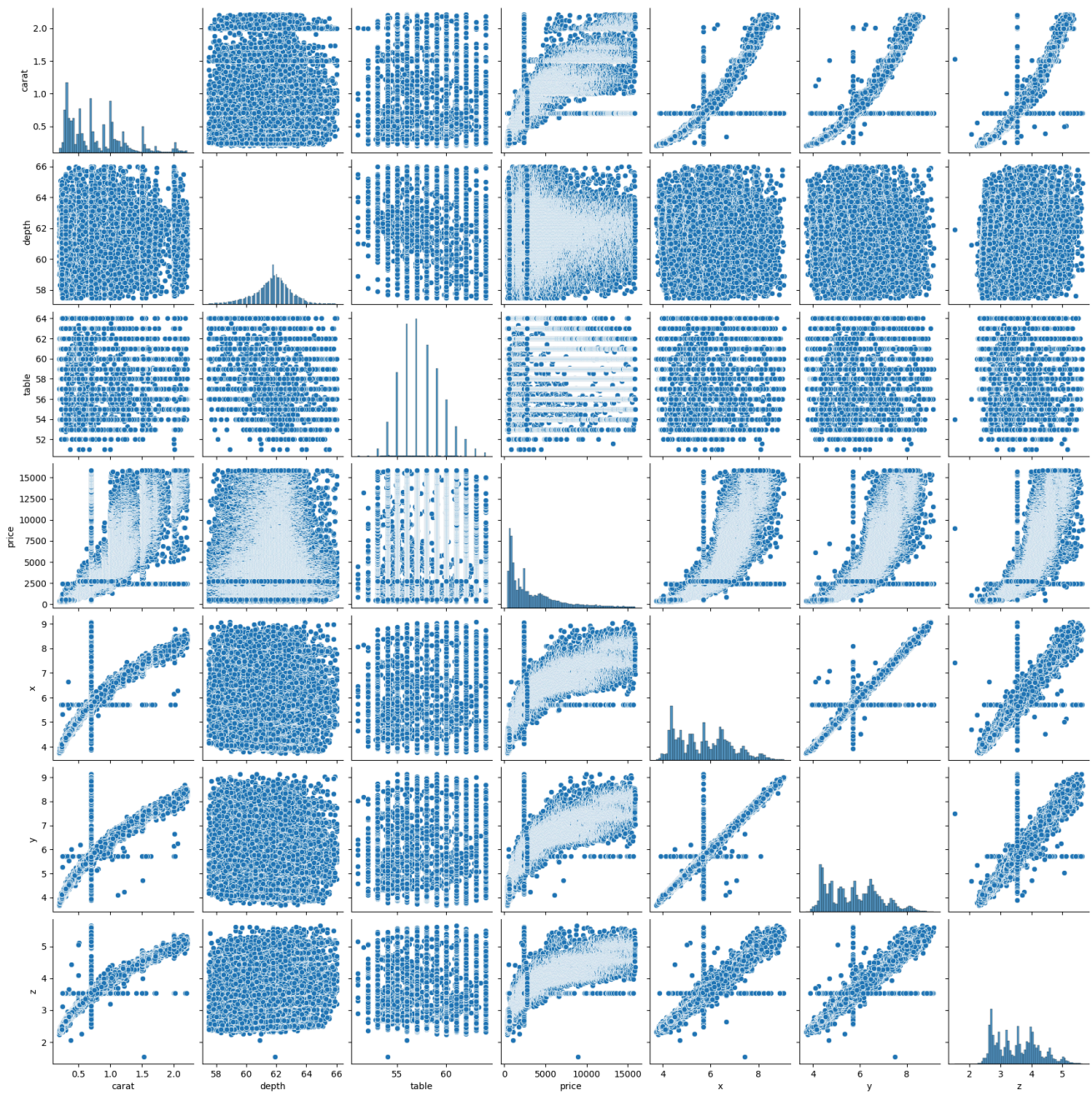

```
# Graph 3 - Box plot of cut vs price
plt.figure(figsize=(10, 6))
sns.boxplot(x='cut', y='price', data=df)
plt.title('Cut vs Price')
plt.xticks(rotation=45)
plt.show()
```




```
# Graph 4 - Histogram of carat distribution
plt.figure(figsize=(10, 6))
sns.histplot(df['carat'], kde=True)
plt.title('Distribution of Carat')
plt.show()
```




```
# Graph 5 - Pair plot of numeric features
sns.pairplot(df[numeric_columns])
plt.show()
```




```
# label encoding
categorical_columns = df.select_dtypes(include=['object']).columns
for column in categorical_columns:
    df[column] = df[column].astype('category').cat.codes


# "Unnamed: 0" to "sno" for clarity
df.rename(columns={"Unnamed: 0": "sno"}, inplace=True)
correlation_with_price = df.corr()['price'].drop('price')
threshold = 0.1
uncorrelated_with_price = correlation_with_price[correlation_with_price.abs() < threshold].index.tolist()

final_columns_to_drop = [
    col for col in uncorrelated_with_price
    if "average" in col or "mined" in col or col == "depth"
]
df.drop(columns=final_columns_to_drop, inplace=True)
print("Columns dropped due to low correlation with price:", final_columns_to_drop)
```

```
Columns dropped due to low correlation with price: ['average us salary', 'number of diamonds mined (millions)', 'depth']
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 49266 entries, 0 to 53939
Data columns (total 10 columns):
```

```
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   sno      49266 non-null  int32
 1   carat    49266 non-null  float64
 2   cut      49266 non-null  int8
 3   color    49266 non-null  int8
 4   clarity  49266 non-null  int8
 5   table    49266 non-null  float64
 6   price    49266 non-null  float64
 7   x        49266 non-null  float64
 8   y        49266 non-null  float64
 9   z        49266 non-null  float64
dtypes: float64(6), int32(1), int8(3)
memory usage: 3.0 MB
```