Result of worksheet2

```
#3.1 Problem1
#1
import pandas as pd
```

```
# Load the bank dataset
df_bank = pd.read_csv("bank (1).csv")

# Display first 5 rows
df_bank.head()
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	unknown	no
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	1	-1	0	unknown	no
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	unknown	no
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	may	92	1	-1	0	unknown	no
4	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	1	-1	0	unknown	no

```
# Display structure and data types
df_bank.info()
```

Show hidden output

```
#2a
# Select columns with datatype 'object'
object_columns = df_bank.select_dtypes(include='object').columns

print("Columns with object datatype:")
print(object_columns)
```

```
Columns with object datatype:
Index(['job', 'marital', 'education', 'default', 'housing', 'loan', 'contact',
       'month', 'poutcome', 'y'],
      dtype='object')
```

```python
#2b
# Loop through object columns and print unique values
for col in object_columns:
    print(f"\nUnique values in column '{col}':")
    print(df_bank[col].unique())
```

```
Unique values in column 'job':
['management' 'technician' 'entrepreneur' 'blue-collar' 'unknown'
 'retired' 'admin.' 'services' 'self-employed' 'unemployed' 'housemaid'
 'student']

Unique values in column 'marital':
['married' 'single' 'divorced']

Unique values in column 'education':
['tertiary' 'secondary' 'unknown' 'primary']

Unique values in column 'default':
['no' 'yes']

Unique values in column 'housing':
['yes' 'no']

Unique values in column 'loan':
['no' 'yes']

Unique values in column 'contact':
['unknown' 'cellular' 'telephone']
```

['no' 'yes']

Unique values in column 'contact':
['unknown' 'cellular' 'telephone']

```
Unique values in column 'month':
['may' 'jun' 'jul' 'aug' 'oct' 'nov' 'dec' 'jan' 'feb' 'mar' 'apr' 'sep']

Unique values in column 'poutcome':
['unknown' 'failure' 'other' 'success']

Unique values in column 'y':
['no' 'yes']
```

```python
#2c
# Count missing values per column
print("Null values in each column:")
print(df_bank.isnull().sum())
```

```
Null values in each column:
age          0
job          0
marital      0
education    0
default      0
balance      0
housing      0
loan         0
contact      0
day          0
month        0
duration     0
campaign     0
pdays        0
previous     0
poutcome     0
y            0
dtype: int64
```

```
#4
# Read the numeric CSV file
df_numeric = pd.read_csv("banknumericdata.csv")

# Display summary statistics
df_numeric.describe()
```

	age	balance	day	duration	campaign	pdays	previous
count	45211.000000	45211.000000	45211.000000	45211.000000	45211.000000	45211.000000	45211.000000
mean	40.936210	1362.272058	15.806419	258.163080	2.763841	40.197828	0.580323
std	10.618762	3044.765829	8.322476	257.527812	3.098021	100.128746	2.303441
min	18.000000	-8019.000000	1.000000	0.000000	1.000000	-1.000000	0.000000
25%	33.000000	72.000000	8.000000	103.000000	1.000000	-1.000000	0.000000
50%	39.000000	448.000000	16.000000	180.000000	2.000000	-1.000000	0.000000
75%	48.000000	1428.000000	21.000000	319.000000	3.000000	-1.000000	0.000000
max	95.000000	102127.000000	31.000000	4918.000000	63.000000	871.000000	275.000000

```
#3.1 Problem2
#1
# Load medical students dataset
df_medical = pd.read_csv("medical_students_dataset.csv")

# Preview data
df_medical.head()
```

	Student ID	Age	Gender	Height	Weight	Blood Type	BMI	Temperature	Heart Rate	Blood Pressure	Cholesterol	Diabetes	Smoking
0	1.0	18.0	Female	161.777924	72.354947	O	27.645835	NaN	95.0	109.0	203.0	No	NaN
1	2.0	NaN	Male	152.069157	47.630941	B	NaN	98.714977	93.0	104.0	163.0	No	No
2	3.0	32.0	Female	182.537664	55.741083	A	16.729017	98.260293	76.0	130.0	216.0	Yes	No
3	NaN	30.0	Male	182.112867	63.332207	B	19.096042	98.839605	99.0	112.0	141.0	No	Yes
4	5.0	23.0	Female	NaN	46.234173	O	NaN	98.480008	95.0	NaN	231.0	No	No

```
#2
```

```python
#2
# Check structure and missing values
df_medical.info()

# Count missing values per column
print(df_medical.isnull().sum())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 13 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   Student ID      180000 non-null  float64
 1   Age             180000 non-null  float64
 2   Gender          180000 non-null  object 
 3   Height          180000 non-null  float64
 4   Weight          180000 non-null  float64
 5   Blood Type      180000 non-null  object 
 6   BMI             180000 non-null  float64
 7   Temperature     180000 non-null  float64
 8   Heart Rate      180000 non-null  float64
 9   Blood Pressure  180000 non-null  float64
 10  Cholesterol     180000 non-null  float64
 11  Diabetes        180000 non-null  object 
 12  Smoking         180000 non-null  object 
dtypes: float64(9), object(4)
memory usage: 19.8+ MB
```

```
Student ID          20000
Age                 20000
Gender              20000
Height              20000
Weight              20000
Blood Type          20000
BMI                 20000
Temperature         20000
Heart Rate          20000
Blood Pressure      20000
Cholesterol         20000
Diabetes            20000
Smoking             20000
dtype: int64
```

```
#3
# Fill numeric columns with mean
numeric_cols = df_medical.select_dtypes(include='number').columns
df_medical[numeric_cols] = df_medical[numeric_cols].fillna(df_medical[numeric_cols].mean())

# Fill categorical columns with mode
categorical_cols = df_medical.select_dtypes(include='object').columns
for col in categorical_cols:
    df_medical[col].fillna(df_medical[col].mode()[0], inplace=True)
```

```
/tmp/ipython-input-3669480358.py:9: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the op


  df_medical[col].fillna(df_medical[col].mode()[0], inplace=True)
```

```
#4
# Check duplicates
print("Duplicate rows:", df_medical.duplicated().sum())

# Remove duplicates
df_medical = df_medical.drop_duplicates()
```

```
Duplicate rows: 12572
```

```
#3.2
# Load Titanic dataset
df_titanic = pd.read_csv("Titanic-Dataset.csv")

df_titanic.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

```
#3.2 Problem1
# Create subset DataFrame
df_subset = df_titanic[['Name', 'Pclass', 'Sex', 'Age', 'Fare', 'Survived']]

# Filter first-class passengers
df_first_class = df_subset[df_subset['Pclass'] == 1]

# Fare statistics
print("Mean Fare:", df_first_class['Fare'].mean())
print("Median Fare:", df_first_class['Fare'].median())
print("Maximum Fare:", df_first_class['Fare'].max())
print("Minimum Fare:", df_first_class['Fare'].min())
```

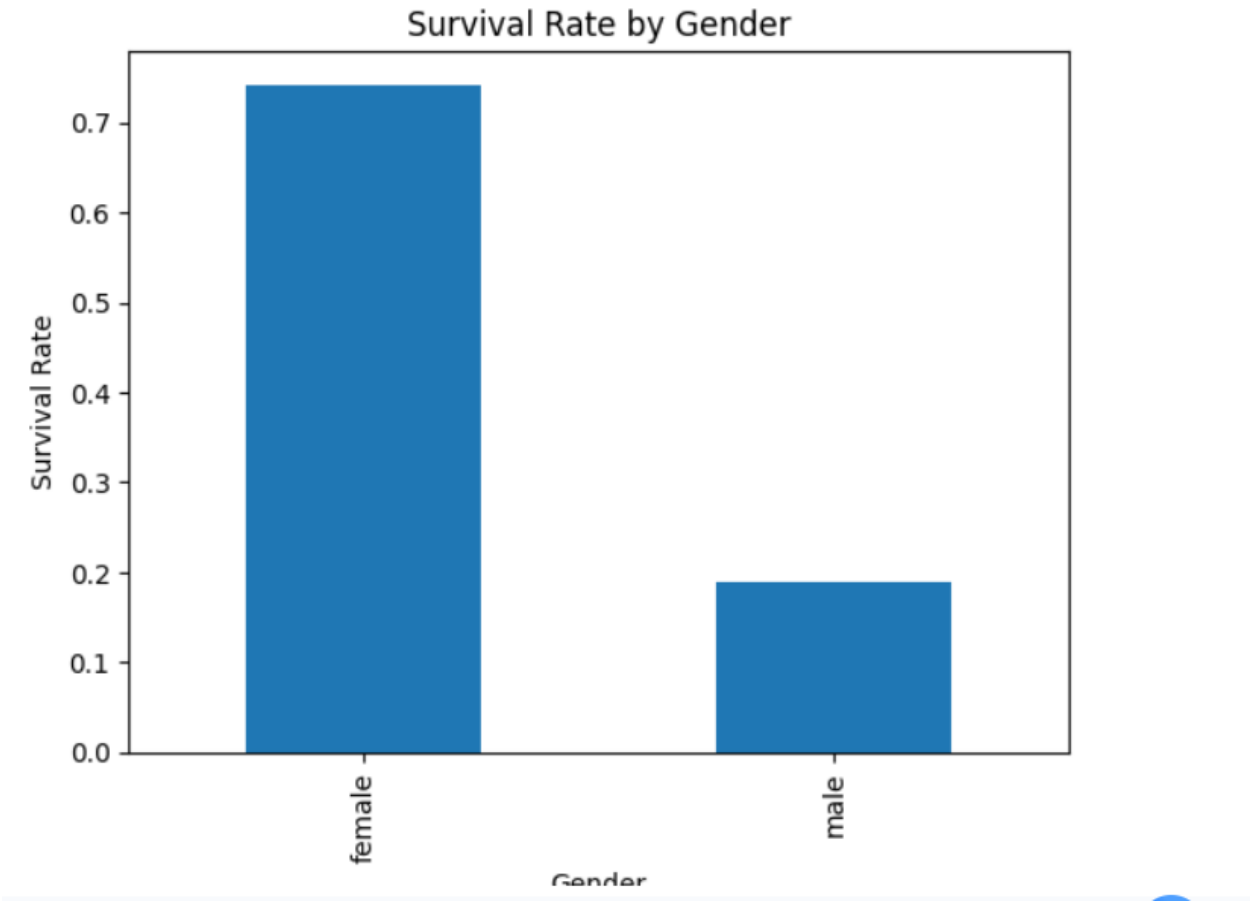
```
Mean Fare: 84.1546875
Median Fare: 60.287499999999994
Maximum Fare: 512.3292
Minimum Fare: 0.0
```

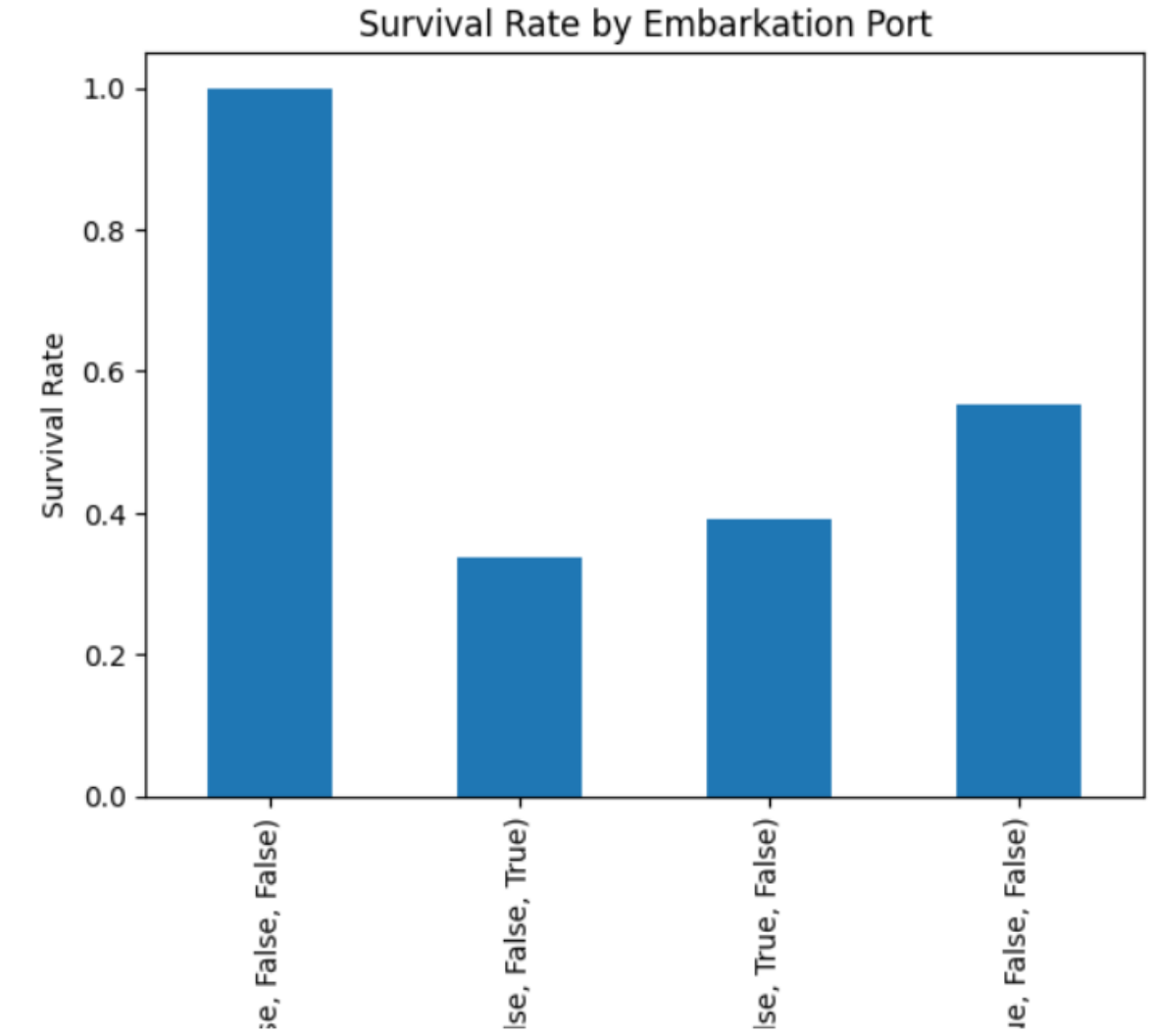
```
#3.2 Problem2
# Count null values in Age
print("Null values in Age:", df_first_class['Age'].isnull().sum())

# Drop rows with missing Age
df_first_class = df_first_class.dropna(subset=['Age'])
```

```
Null values in Age: 30
```

```python
#3.2 problem3
# One-hot encode Embarked column
embarked_dummies = pd.get_dummies(df_titanic['Embarked'], prefix='Embarked')

# Add encoded columns
df_titanic = pd.concat([df_titanic, embarked_dummies], axis=1)

# Drop original Embarked column
df_titanic.drop(columns=['Embarked'], inplace=True)

# Verify changes
df_titanic.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked_C	Embarked_Q	Embarked_S
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	False	False	True
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	True	False	False
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	False	False	True
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	False	False	True
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	False	False	True





Terminal

(False, Fals
(False, Fals
(False, Tru
(True, Fals

Embarked_C,Embarked_Q,Embarked_S