PAI ASSIGNMENT 2

Q1:

```python
import requests

def getLiveScores():
    print("Fetching Live Cricket Scores...")
    with open("api.txt","r") as file:
        apiUrl = file.read()
    try:
        response = requests.get(apiUrl)
        if(response.status_code!=200):
            print(f"HTTP Error:{response.status_code}")
            print(f"Raw Response:{response.text}")
            return
        data = response.json()
        for key , value in data.items():
            if key != "data":
                print(f"{key}:{value}")
        print()
        if data.get("status") != "success":
            reason = data.get("API returned failure status")
            print(f"API Error:{reason}")
            return
        if "data" not in data or not data["data"]:
            print("No matches found")
        matchList = data["data"]
        print(f"Found {len(matchList)} matches.")
```
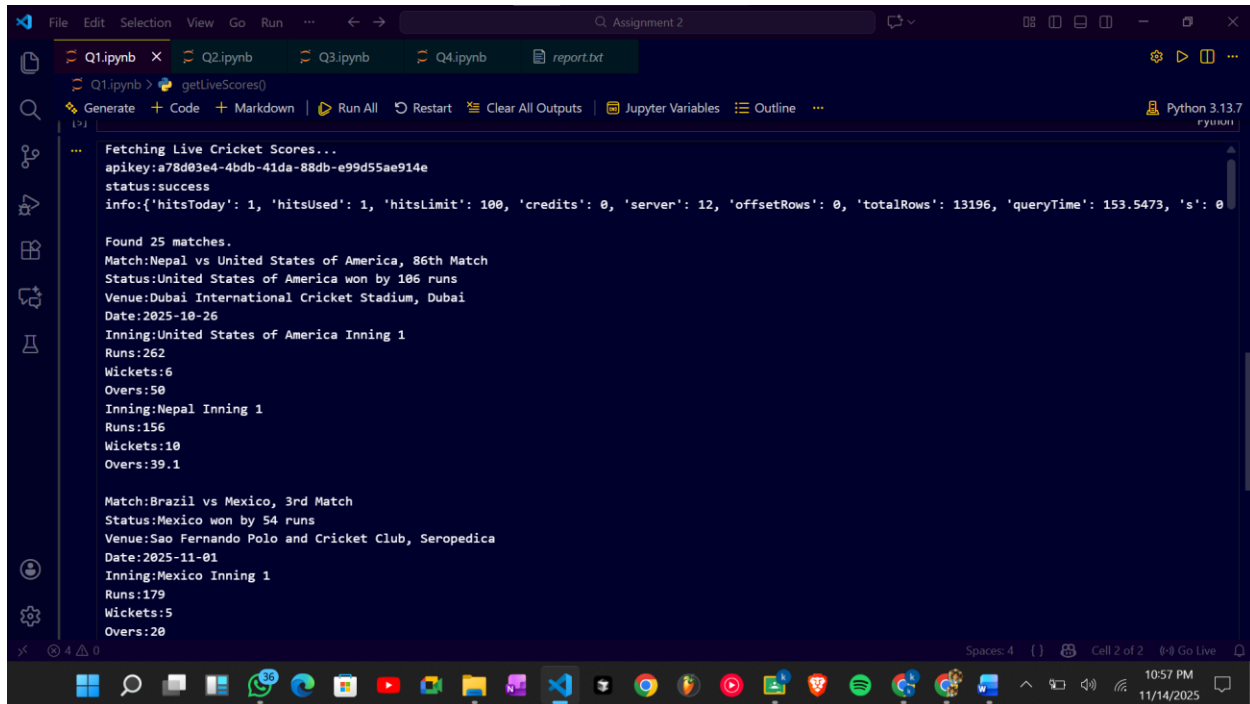
```python
    for match in matchList:

        print(f"Match:{match.get('name','N/A')}")

        print(f"Status:{match.get('status','N/A')}")

        print(f"Venue:{match.get('venue','N/A')}")

        print(f"Date:{match.get('date','N/A')}")

        if "score" in match and match["score"]:

            for inning in match["score"]:

                print(f"Inning:{inning.get('inning','N/A')}")

                print(f"Runs:{inning.get('r','N/A')}")

                print(f"Wickets:{inning.get('w','N/A')}")

                print(f"Overs:{inning.get('o','N/A')}")

        else:

            print("No score available.")

        print()

    except requests.exceptions.RequestException as e:

        print(f"Error:{e}")

getLiveScores()
```

Output:

Q2:

```python
import numpy as np

my_list = []

with open("sensor_data.csv","r") as file:

    for item in file:

        row = item.strip().split(",")

        rowFloat = [float(value) for value in row]

        my_list.append(rowFloat)

Array = np.array(my_list)

Array[Array == -999] = np.nan

Array[(Array<0) | (Array>100)] = np.nan

columnsMean = np.nanmean(Array,axis=0)

rowMedian = np.nanmedian(Array,axis=1)

nanCount = np.sum(np.isnan(Array),axis=0)
```

```
sensorMostInvalid = np.argmax(nanCount)

maxInvalidCount = nanCount[sensorMostInvalid]

maxVal = np.nanmax(Array)

minVal = np.nanmin(Array)

dataNormalised = (Array - minVal) / (maxVal - minVal)

np.savetxt("sensor_data_normalized.csv",dataNormalised,delimiter=",",fmt='%.6f')
```

Output:

Q3:

```python
import pandas as pd

df = pd.read_csv("Titanic-Dataset.csv")

reportLines = []

for column in df.columns:

    dataType = df[column].dtype

    missingPart = df[column].isna().mean() * 100

    reportLines.append(f'{column}: {dataType}, missing:{missingPart}')
```

```python
with open("inspect_report.txt","w") as file:

    for line in reportLines:

        file.write(line+"\n")

df["Age"] = df.groupby(['Pclass','Sex'])["Age"].transform(lambda g : g.fillna(g.median()))

embarkedMode = df["Embarked"].mode()[0]

df["Embarked"] = df["Embarked"].fillna(embarkedMode)

if("Cabin" in df.columns):

    df = df.drop(columns=["Cabin"])

df['FamilySize'] = df["SibSp"] + df["Parch"]

df["IsAlone"] = (df["FamilySize"] == 0).astype(int)

df["Age"] = df["Age"].astype('int64')

df.to_csv("titanic_cleaned.csv",index=False)
```
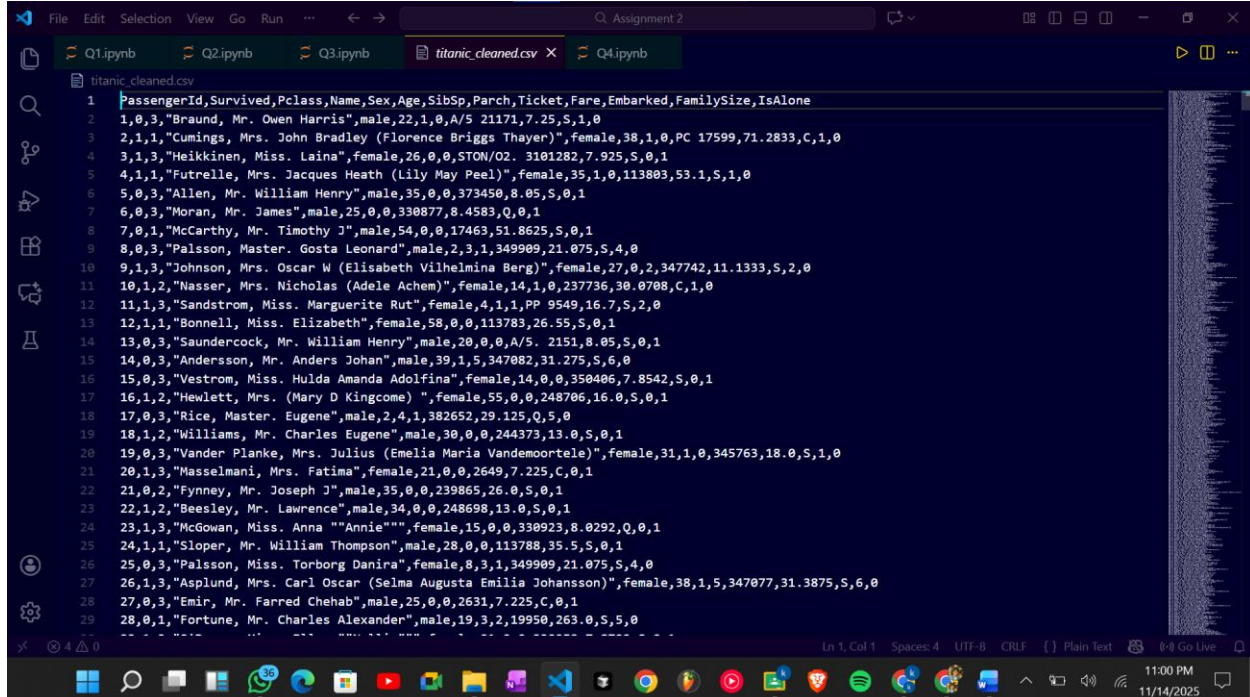
Output:

Q4:

```python
import pandas as pd

titanicDf = pd.read_csv("titanic_cleaned.csv")

faresDf = pd.read_csv("ticket_fares.csv")

df = pd.merge(titanicDf,faresDf,on="Ticket",how="left")

bins = [0,12,19,30,61,120]

labels = ['Child','Teenager','Young Adult','Adult','Senior']

df['AgeGroup'] = pd.cut(df['Age'],bins = bins,labels = labels,include_lowest=True)

survivedGenderAge = df.groupby(['Age','AgeGroup'],observed=False)['Survived'].mean().reset_index()

survivedPClass = df.groupby("Pclass")["Survived"].mean().reset_index()

df["FareBin"] = pd.qcut(df['Fare_x'] , q = 4 , labels = ["Low","Medium","High","VeryHigh"])

survivedFareBin = df.groupby("FareBin",observed=False)["Survived"].mean().reset_index()

paragraph1 = """Survival rates grouped by gender and age indicate that women and
children had higher chances of survival compared to men, adults, and seniors.

This supports the idea that women and children were given priority during evacuation."""


paragraph2 = """Passengers in 1st Class had significantly higher survival rates than those in
2nd and 3rd Class.

When examining fare-based quantile bins, survival rates also rise with increasing fares.

This indicates that wealth was strongly linked to a passenger's chances of survival."""

with open('report.txt','w') as file:

    file.write(paragraph1.strip() + "\n" + paragraph2.strip())
```
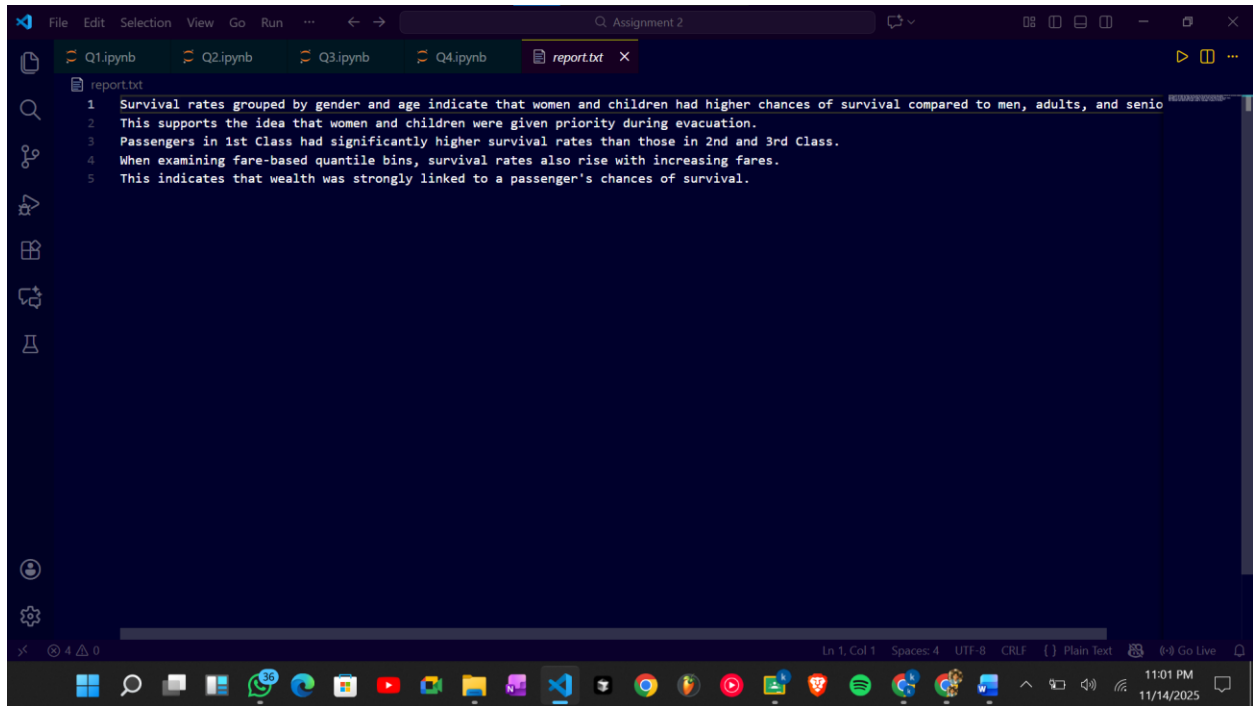
Output:

report.txt

```
1  Survival rates grouped by gender and age indicate that women and children had higher chances of survival compared to men, adults, and senio
2  This supports the idea that women and children were given priority during evacuation.
3  Passengers in 1st Class had significantly higher survival rates than those in 2nd and 3rd Class.
4  When examining fare-based quantile bins, survival rates also rise with increasing fares.
5  This indicates that wealth was strongly linked to a passenger's chances of survival.
```