Question 1:

```
import requests
import json
def getLiveScores():
    with open('api.txt','r') as f:
        apiUrl = f.read().strip()
    if not apiUrl:
        print("API URL is empty! Please check 'api.txt'")
        return


    try:
        response = requests.get(apiUrl)
        if response.status_code == 200:
            apiData = response.json()


            info = apiData.get("info")
            if info:
                for key, value in info.items():
                    print(f"{key}: {value}")



            if apiData.get("status") == "success" and "data" in apiData and
apiData['data']:
                matchList = apiData['data']
                print(f"\nFound {len(matchList)} matches.\n")


                for match in matchList:
```

```
                    if "score" in match and match["score"]:
                        print(f"Match: {match['name']}")
                        print(f"Status: {match['status']}")
                        print(f"Venue: {match['venue']}")
                        print(f"Date: {match['date']}")

                        for entry in match["score"]:
                            print(f"Innings: {entry['inning']}")
                            print(f"Runs: {entry['r']}")
                            print(f"Wickets: {entry['w']}")
                            print(f"Overs: {entry['o']}\n")
                    else:
                         print(f"Match: {match['name']}")
                         print(f"Status: {match['status']}")
                         print(f"Venue: {match['venue']}")
                         print(f"Date: {match['date']}")



            else:
                errorMsg = apiData.get("reason","No live matches found or
unknown error occurred.")
                print(f"Couldn't receive matches: {errorMsg}")


        else:
            print(f"Error: Failed to fetch data. Status Code:
{response.status_code}")
            print("Response:", response.text)
```

```
    except requests.exceptions.RequestException as e:
        print(f"A network error occurred: {e}")
getLiveScores()
```

```
...   hitsToday: 5
      hitsUsed: 1
      hitsLimit: 100
      credits: 0
      server: 18
      offsetRows: 0
      totalRows: 10
      queryTime: 30.4446
      s: 0
      cache: 1

      Found 10 matches.

      Match: Thailand Women vs Namibia Women, 2nd Match
      Status: Thailand Women won by 39 runs
      Venue: Terdthai Cricket Ground, Bangkok
      Date: 2025-11-14
      Innings: thailand women Inning 1
      Runs: 124
      Wickets: 5
      Overs: 20

      Innings: Thailand Women,Namibia Women Inning 1
      Runs: 85
      Wickets: 10
      ...
      Runs: 37
      Wickets: 1
      Overs: 20
```

Question 2:

```
import numpy as np
with open('sensor_data.csv','r') as f:
    lines= f.readlines()
```

```
dataList = [line.strip().split(',') for line in lines]
dataArray = np.array(dataList, dtype=float)
dataArray[dataArray == -999] = np.nan
mask = (dataArray < 0) | (dataArray > 100)
dataArray[mask] = np.nan
dataArray
mean_per_sensor = np.nanmean(dataArray,axis=1)
mean_per_hour = np.nanmedian(dataArray,axis = 0)
null_count = np.isnan(dataArray).sum(axis=1)

X_min = np.nanmin(dataArray)
X_max = np.nanmax(dataArray)
X_norm = (dataArray - X_min)/(X_max - X_min)
X_norm
np.savetxt("sensor_data_normalized.csv",X_norm,delimiter=",")
```

```
array([[0.140125, 0.71075 , 0.471625, ..., 0.632875, 0.006375, 0.91975 ],
       [0.4655  , 0.77125 , 0.7305  , ..., 0.736375, 0.246   , 0.359   ],
       [0.363375, 0.148125, 0.27325 , ..., 0.034625, 0.02975 , 0.08125 ],
       ...,
       [0.1075  , 0.686125, 0.39875 , ..., 0.081125, 0.2755  , 0.337125],
       [0.150375, 0.41575 , 0.570125, ..., 0.0445  , 0.921125, 0.646   ],
       [0.43825 ,      nan, 0.129625, ..., 0.954   , 0.312875, 0.30925 ]],
      shape=(720, 100))
```

Question 3:

```
import pandas as pd
df = pd.read_csv("Titanic-Dataset.csv")
df.info()
```

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

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

df.drop(columns=["Cabin"],inplace=True)

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

df["IsAlone"] = df["FamilySize"].apply(lambda x:1 if x==0 else 0)

df["Age"] = df["Age"].astype(int)

survived = df[df["Survived"] == 1]

survived= survived["Survived"].value_counts()

survived

df.to_csv("titanic_cleaned.csv",sep= ',',index= False)

print("Data saved successfully!")

df
```

```
Data saved successfully!
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	FamilySize	IsAlone
0	1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.2500	S	1	0
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38	1	0	PC 17599	71.2833	C	1	0
2	3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.9250	S	0	1
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1000	S	1	0
4	5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.0500	S	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27	0	0	211536	13.0000	S	0	1
887	888	1	1	Graham, Miss. Margaret Edith	female	19	0	0	112053	30.0000	S	0	1
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	21	1	2	W./C. 6607	23.4500	S	3	0
889	890	1	1	Behr, Mr. Karl Howell	male	26	0	0	111369	30.0000	C	0	1
890	891	0	3	Dooley, Mr. Patrick	male	32	0	0	370376	7.7500	Q	0	1

891 rows × 13 columns

Question 4:

```
import pandas as pd

import numpy as np

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

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

pd.merge(tdf,fdf,on='Ticket')

ageBins =[0,12,19,59,100]
```

```
ageLabels= ['Child','Young Adult','Adult','Senior']

tdf["AgeGroup"] = pd.cut(tdf["Age"],bins=
ageBins,labels=ageLabels,right=True)

tdf.groupby(['Sex','AgeGroup'])['Survived'].mean()

with open("report.txt","w") as f:

    f.write("Hypothesis A\n")

    f.write("The numbers clearly show that women and children were given
priority during the Titanic rescue. Most women survived, with survival rates
between 74% to 100%, while only about 10% to 17%  men survived. Children also
had better chances of survival, with around 50% to 57% were saved. ")

tdf.groupby("Pclass")["Survived"].mean()

fareLabels = ['Low','Medium','High','VeryHigh']

tdf["FareBin"] = pd.qcut(tdf["Fare"],q=4,labels=fareLabels)

tdf.groupby("FareBin")["Survived"].mean()

with open("report.txt","a") as f:

    f.write("\nHypothesis B\n")

    f.write("The data clearly shows that wealthier passengers had a higher
chance of survival on the Titanic. Passengers in the lowest fare group had
only about 19% survival, while those in the highest fare group had around
58%. This suggests that people who paid more, likely traveling in higher
classes, were given better access to lifeboats and safety, showing a clear
link between wealth and survival.")

tdf
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	FamilySize	IsAlone	AgeGroup	FareBin
0	1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.2500	S	1	0	Adult	Low
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38	1	0	PC 17599	71.2833	C	1	0	Adult	VeryHigh
2	3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.9250	S	0	1	Adult	Medium
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1000	S	1	0	Adult	VeryHigh
4	5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.0500	S	0	1	Adult	Medium
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27	0	0	211536	13.0000	S	0	1	Adult	Medium
887	888	1	1	Graham, Miss. Margaret Edith	female	19	0	0	112053	30.0000	S	0	1	Young Adult	High
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	21	1	2	W./C. 6607	23.4500	S	3	0	Adult	High
889	890	1	1	Behr, Mr. Karl Howell	male	26	0	0	111369	30.0000	C	0	1	Adult	High
890	891	0	3	Dooley, Mr. Patrick	male	32	0	0	370376	7.7500	Q	0	1	Adult	Low

891 rows × 15 columns