```
import pandas as pd


 #load Excel file

df = pd.read_excel("marksheet (1).xlsx")

 #choose subject columns (skip Name)

subject_cols = df.columns[1:]


# convert all subject columns to numeric (non-numbers become NaN)

df[subject_cols] = df[subject_cols].apply(pd.to_numeric, errors='coerce')


#calculate average ignoring NaN

df['Average'] = df[subject_cols].mean(axis=1)


#show the result

print("\n--- Data with averages ---")

print(df)


#find top student

top_student = df.loc[df['Average'].idxmax()]


print("\n--- Top Student ---")

print("Name:", top_student['Name'])

print("Average Score:", top_student['Average'])

# ====== SETTINGS ======

FILE_PATH = "marksheet (1).xlsx"  # put your file path here
```

```
def analyze():
  try:
    # Read Excel file
    df = pd.read_excel(FILE_PATH)


    # Subject columns (skip Name)
    subject_cols = df.columns[1:]
    # Convert to numeric
    df[subject_cols] = df[subject_cols].apply(pd.to_numeric, errors='coerce')
    # Average
    df['Average'] = df[subject_cols].mean(axis=1)


    # Top & bottom
    top_student = df.loc[df['Average'].idxmax()]
    bottom_student = df.loc[df['Average'].idxmin()]


    # Clear old tree content
    for i in tree.get_children():
      tree.delete(i)
    # Insert rows
    for _, row in df.iterrows():
      tree.insert("", tk.END, values=list(row))


    # Pop-up message
    messagebox.showinfo(
      "Results",
      f"Top Student: {top_student['Name']} (Avg: {top_student['Average']:.2f})\n"
      f"Lowest Student: {bottom_student['Name']} (Avg: {bottom_student['Average']:.2f})"
    )

    # Graphs
    plt.figure(figsize=(6,4))
    plt.bar(subject_cols, top_student[1:len(subject_cols)+1])
    plt.title(f"Highest Scoring Student: {top_student['Name']}")
    plt.xlabel("Subjects")
    plt.ylabel("Marks")
    plt.show()

    plt.figure(figsize=(6,4))
    plt.bar(subject_cols, bottom_student[1:len(subject_cols)+1], color='orange')
    plt.title(f"Lowest Scoring Student: {bottom_student['Name']}")
    plt.xlabel("Subjects")
    plt.ylabel("Marks")
    plt.show()

  except Exception as e:
    messagebox.showerror("Error", str(e))

# ========== GUI ==========
root = tk.Tk()
root.title("Student Marks Analyzer")
```

```
# Background color
root.configure(bg="#e6f7ff")

title_label = tk.Label(root, text="Student Marks Analyzer",
                 font=("Arial", 16, "bold"), bg="#3399ff", fg="white")
title_label.pack(fill="x", pady=5)

btn = tk.Button(root, text="Analyze Marks", font=("Arial", 12, "bold"),
            bg="#66ccff", fg="black", command=analyze)
btn.pack(pady=10)

# Treeview to show dataframe
frame = tk.Frame(root)
frame.pack(fill="both", expand=True, padx=10, pady=10)

tree = ttk.Treeview(frame, show="headings")
tree.pack(fill="both", expand=True)

# Set up tree columns after reading file once to know headers
try:
   df_preview = pd.read_excel(FILE_PATH)
   subject_cols_preview = df_preview.columns[1:]
   df_preview[subject_cols_preview] =
df_preview[subject_cols_preview].apply(pd.to_numeric, errors='coerce')
   df_preview['Average'] = df_preview[subject_cols_preview].mean(axis=1)
   cols = list(df_preview.columns)
```

```
    tree["columns"] = cols
    for col in cols:
        tree.heading(col, text=col)
        tree.column(col, width=100, anchor="center")
except Exception as e:
    messagebox.showerror("Error", f"Could not preview file: {e}")


root.geometry("800x400")
root.mainloop()
import pandas as pd
import matplotlib.pyplot as plt
import tkinter as tk
from tkinter import ttk, messagebox

# ====== SETTINGS ======
FILE_PATH = "marksheet (1).xlsx"  # put your file path here

def analyze():
    try:
        # Read Excel file
        df = pd.read_excel(FILE_PATH)

        # Subject columns (skip Name/Section)
        subject_cols = df.columns[1:]  # assumes first col is Name

        # Convert to numeric (errors to NaN)
        df[subject_cols] = df[subject_cols].apply(pd.to_numeric, errors='coerce')

        # Class average per subject (mean across all students)
        subject_avgs = df[subject_cols].mean()

        # Clear tree
        for i in tree.get_children():
            tree.delete(i)

        # Show averages in table
        for subject, avg in subject_avgs.items():
            tree.insert("", tk.END, values=(subject, round(avg, 2)))

        # Pop-up
        messagebox.showinfo("Class Averages", "Calculated average marks per subject")

        # Plot bar chart
        plt.figure(figsize=(6,4))
        plt.bar(subject_avgs.index, subject_avgs.values, color="#3399ff")
        plt.title("Average Marks per Subject")
        plt.xlabel("Subjects")
        plt.ylabel("Average Marks")
        plt.show()

    except Exception as e:
        messagebox.showerror("Error", str(e))
```

```
# ========== GUI ==========
root = tk.Tk()
root.title("Subject Average Marks")
root.configure(bg="#e6f7ff")

title_label = tk.Label(root, text="Subject Average Marks",
                       font=("Arial", 16, "bold"), bg="#3399ff", fg="white")
title_label.pack(fill="x", pady=5)

btn = tk.Button(root, text="Calculate Averages", font=("Arial", 12, "bold"),
                bg="#66ccff", fg="black", command=analyze)
btn.pack(pady=10)

frame = tk.Frame(root)
frame.pack(fill="both", expand=True, padx=10, pady=10)

tree = ttk.Treeview(frame, show="headings")
tree["columns"] = ("Subject", "Average Marks")
tree.heading("Subject", text="Subject")
tree.heading("Average Marks", text="Average Marks")
tree.column("Subject", width=150, anchor="center")
tree.column("Average Marks", width=150, anchor="center")
tree.pack(fill="both", expand=True)

root.geometry("400x300")
root.mainloop()
```