PokeDex Tkinter Python and Pandas Assignment

In this assignment you will be creating a PokeDex application using Python and Pandas.

A Pokédex is a digital encyclopaedia in the Pokémon universe that provides detailed information about different Pokémon species. It is a crucial tool for Pokémon Trainers, helping them learn about the various Pokémon they encounter and capture.

Key Features of a Pokédex:

- Species Information: Includes data on each Pokémon's type, abilities, and evolutionary stages.
- Pokémon Stats: Provides information on a Pokémon's strengths, weaknesses, and battle statistics.
- Appearance: Displays the Pokémon's image.

You have been given a detailed dataset to work with and you are required to create a Pokedex application using Python, Pandas, Pillow and Matplotlib. This assignment is multi parts. Each part will contain its own marks, and you will be assessed on how well you have done individually.

"IMPORTANT #1: DOWNLOAD RESOURCES ASSOCIATED WITH THIS ASSIGNMENT ON TEAMS. "

"IMPORTANT #2: THE CODE AND COMMENT YOU WRITE IN THE PROGRAM MUST BE YOUR OWN, EACH ASSIGNMENT WILL BE COMPARED FOR SIMILARITIES THEREFORE YOU WILL LOSE MARKS IF THERE ARE MULTIPLE SIMILARITIES"

Pokémon Images Folder	**1,025 files images of Pokémon's**
Pokémon dataset	Pokémon Excel File
Pokémon card template	A JPG file
Pokémon types folder	18 Pokémon types image
NOVA custom front file	NOVA ttf file (optional)
Sample Scripts	I've included a few sample scripts to get you started on the assignment. Use them wisely.

Now you have access to the files and folders from above, you will need to create your own project inside of PyCharm and set up the files and folders to communicate with your Python Script.

Task 1:

Objective: Use the provided user interface (UI) to create a functional Pokédex application.

Steps:

1. **Read the Dataset**: Your application should read the Pokémon dataset provided.
2. **Populate the List**: Use Tkinter to create a list that displays only the names of the Pokémon.

3. **Display Pokémon Information**: When a Pokémon name is selected from the list, the application should display the following details for that Pokémon:
    - Name
    - HP
    - Attack
    - Defence
    - Speed
    - Special Attack (SP. Attack)
    - Special Defence (SP. Defence)
    - Total (sum of all stats)
4. **Show Pokémon Image**: Ensure that selecting a Pokémon name also displays the corresponding image from the provided folder.

Requirements:

- **User Interface**: Utilize Tkinter to build the UI components.
- **Data Handling**: Use Pandas to read and manage the dataset.
- **Image Display**: Integrate the images correctly so that each Pokémon's image appears when their name is selected.

Tips:

- Make sure your UI is user-friendly and intuitive.
- Ensure that the data displayed is accurate and matches the selected Pokémon.
- Test your application thoroughly to handle different scenarios and inputs.

Task 2:

Objective: Create charts to visualize the top 5 Pokémon in various categories using Matplotlib.

Steps:

1. **Read the Dataset**: Use Pandas to read the Pokémon dataset provided.
2. **Identify Top and Bottom Pokémon**: For each category listed below, identify the top 5 Pokémon with the highest values and the bottom 5 Pokémon with the lowest values.
3. **Create Charts**: Use Matplotlib to create charts that clearly display the top 5 Pokémon for each category. Ensure each chart is labelled correctly and is easy to understand.

Categories [Pick any 4]:

1. **Highest HP**: Create a chart showing the top 5 Pokémon with the highest HP (Hit Points).
2. **Highest Defense**: Create a chart showing the top 5 Pokémon with the highest Defense.
3. **Highest Speed**: Create a chart showing the top 5 Pokémon with the highest Speed.

4. **Highest Total**: Create a chart showing the top 5 Pokémon with the highest Total (sum of all stats).
5. **Lowest HP**: Create a chart showing the top 5 Pokémon with the lowest HP.
6. **Lowest Defense**: Create a chart showing the top 5 Pokémon with the lowest Defense.
7. **Lowest Speed**: Create a chart showing the top 5 Pokémon with the lowest Speed.
8. **Lowest Total**: Create a chart showing the top 5 Pokémon with the lowest Total (sum of all stats).

Requirements:

- **Data Handling**: Use Pandas to filter and sort the dataset to find the top and bottom Pokémon for each category.
- **Chart Creation**: Use Matplotlib to create clear and informative charts. Each chart should include:
    - A title that describes the category (e.g., "Top 5 Pokémon with Highest HP").
    - Labels for the x-axis (Pokémon names) and y-axis (stat values).
    - Appropriate colours and legends to make the charts visually appealing.

Tips:

- Ensure your charts are easy to read and interpret.
- Use different colours or styles to distinguish between the top and bottom categories.
- Test your charts to make sure they accurately represent the data.

Example: For the "Top 5 Pokémon with Highest HP" chart, you might create a bar chart with Pokémon names on the x-axis and their HP values on the y-axis. Each bar would represent one of the top 5 Pokémon in this category.

You can add these actions as buttons in the PokeDex UI for ease of use. No terminal or command prompt interaction is allowed in this assignment.

Task 3:

Dark	Dragon	Electric	Fairy	Fighting	Fire
Flying	Ghost	Grass	Ground	Ice	Normal
Poison	Psychic	Rock	Steel	Water	Bug

1. **Create a New Column**: In the dataset, create a new column named "Types". Use Pandas and Python to randomly assign one of the 18 types to each Pokémon entry.

2. **Display Type Images**: Use Tkinter to display the type images in the UI. Ensure that the correct image is loaded and displayed for each Pokémon type. You can place the image anywhere on the screen, but it must accurately reflect the type assigned in the dataset.

Requirements:

- **Data Handling**: Use Pandas to manipulate the dataset and assign types.
- **Random Assignment**: Ensure the types are assigned randomly to each Pokémon.
- **UI Integration**: Use Tkinter to display the type images correctly in the user interface.

Tips:

- Verify that each Pokémon entry has a type assigned and that the corresponding image is displayed correctly.
- Make sure the UI is clear and the type images are easily identifiable.
- Test your application to ensure the type images match the assigned types in the dataset.

Task 4:

Objective: Add a feature to your program that generates Pokémon cards as images.

Steps:

1. **Create Pokémon Cards**: Implement a feature in your program that generates an image of each Pokémon using the provided information.
2. **Use the Card Template**: Populate the card template with the necessary details (e.g., Name, HP, Attack, Defense, Speed, Special Attack, Special Defense, and Type).
3. **Save as JPG**: Ensure the program saves each generated card as a JPG file in the same directory as your script.
4. **Use Pillow**: Utilize the Python Pillow framework to create and manipulate the images based on the provided information.

Requirements:

1. **Card Template**: Use the provided card template to ensure consistency in design.
2. **Information Accuracy**: Make sure all the Pokémon information is correctly displayed on the card.
3. **File Saving**: The program should save each card as a JPG file in the correct directory.

Tips:

- Verify that the information on each card is accurate and matches the data in your dataset.
- Ensure the card design is visually appealing and easy to read.
- Test the feature to confirm that the cards are generated and saved correctly.

Good luck and have fun creating your Pokémon cards! 📊 🐾

File Submission [Deadline 18/12/2024]

Screen Recording (**Video File**) –

1. Record your application working and demonstrate each of the following features
    - Application is reading the dataset
    - From the list box you can select the Pokemon and display its stats and image
    - UI has been placed in the background
    - Appropriate fonts have been used
    - Multiple charts are displaying
    - Pokemon Card is being created and saved to the disk
2. Full commented source code (**.py**)
    - Code is error free
    - Comments are meaningful and detailed
    - Appropriate functions and variable names are used
    - Loops and lists are used
    - Reading and writing data
    - Loading files from folders
    - Appropriate

    DO NOT SUBMIT THE IMAGES OR THE DATASET, YOUR CODE WILL BE RAN ON THE DEFAULT DATASET ON ANOTHER SYSTEM.

Marking Grid:

Understanding of the Project and Use of Pandas and Tkinter (20%)

- **Project Requirements**: Assess how well the student understands what the project is asking for. This includes their grasp of the objectives, the expected outcomes, and the specific tasks they need to complete.
- **Pandas**: Evaluate the student's ability to use Pandas for data manipulation and analysis. This includes tasks like reading data from files, cleaning and transforming data, and performing operations like grouping and aggregating.
- **Tkinter**: Look at how effectively the student uses Tkinter to create the graphical user interface (GUI). This includes the layout, widgets used, and how user interactions are handled.

Originality of Work (20%)

- **Uniqueness**: Assess how original the student's solution is. This means checking if they have come up with their own ideas and approaches rather than copying from existing projects.
- **Inspiration vs. Plagiarism**: While it's okay to draw inspiration from other sources, the student should demonstrate their own understanding and creativity. Their work should reflect their personal touch and problem-solving skills.

Innovation, Creativity, and Flair (15%)

- **Creative Solutions**: Look at how creatively the student has approached the project. This could include unique features, interesting ways of solving problems, or innovative uses of the tools and libraries.
- **Problem-Solving**: Evaluate how the student has tackled challenges and obstacles in the project. Have they come up with clever or efficient solutions?
- **Overall Flair**: Consider the overall impression of the project. Does it stand out due to its creativity and innovation?

Code Comments and Documentation (15%)

- **Detailed Comments**: Check if the student has provided clear and detailed comments in their code. These comments should explain what the code is doing and why certain decisions were made.
- **Understanding**: The comments should reflect a deep understanding of the code. They should help someone else (or the student themselves later) understand the logic and flow of the program.

Producing a Pokémon Card (15%)

- **Data Usage**: Assess how well the program uses the data to generate a Pokémon card. This includes pulling the correct information and displaying it accurately.
- **Design**: Evaluate the design of the Pokémon card. Is it visually appealing and well-organized? Does it include all the necessary information (e.g., name, type, stats)?
- **Functionality**: Check if the Pokémon card feature works smoothly without errors. The card should be generated correctly for different Pokémon.

Making Charts Using Matplotlib (15%)

- **Informative Charts**: Evaluate the student's ability to create charts that effectively communicate information. The charts should be relevant to the project and provide valuable insights.
- **Visual Appeal**: The charts should be visually appealing, with appropriate use of colours, labels, and legends. They should be easy to read and understand.
- **Technical Proficiency**: Assess the student's proficiency with Matplotlib. This includes their ability to customize charts, handle different types of data, and integrate the charts into the overall project.

If you score higher than 55% on your assignment, you will receive a personalised PokeDex completion certificate. It will be printed on certificate grade paper. Display your name and overall score. Do your best.