Announcement for Premium Batch 1 ECO Techno Leaders (Batch 9 and Batch 10)

Subject: Submission of Assessment Task for Green Finance Data Analysis

Date: 29 June 2025

Dear Participants of Premium Batch 1 ECO Techno Leaders (Batch 9 and Batch 10),

We are pleased to announce the assessment task for your programme, designed to evaluate your proficiency in Python programming and its application to real-world data analysis within the context of Green Finance and renewable energy projects in Indonesia for 2024. This task offers a significant opportunity to demonstrate your technical and analytical capabilities, contributing to Indonesias sustainable development objectives under the Green Finance initiative.

Task Description

You are required to complete seven examination questions, supplemented by one bonus question, based on the provided Green Finance datasets. These questions assess intermediate-level Python skills from Chapters 5 to 11, encompassing:

- Conditional statements (if-else)
- Loops (for and while)
- Lists and dictionaries
- Functions and modules
- Error handling

The bonus question involves applying Machine Learning/AI techniques using a Decision Tree Classifier. The datasets provided are:

- Economic_Dataset.xlsx
- Social_Dataset.xlsx
- Environmental_Dataset.xlsx
- Geospatial_Dataset.xlsx
- Financial_Dataset.xlsx

Dataset Explanations

To facilitate your understanding of the datasets, the following sections provide descriptions of key fields relevant to the exam questions, translated and adapted from the programmes tutorial resources.

Economic Dataset

- **GDP_Growth**: Annual GDP growth rate (in percent). Higher growth supports demand for energy and renewable projects.

- **Daya_Tarik_Investasi**: Investment attractiveness (High, Medium, Low). Indicates the economic appeal of the projects region.

Social Dataset

- **Land_Status**: Land ownership type (Adat, Negara, Swasta). Influences project feasibility and social conflict risk.
- **Tingkat_Konflik**: Social conflict level (High, Medium, Low). Reflects community acceptance and potential risks.

Environmental Dataset

- **CO2_Reduction**: Annual CO2 emissions reduction (in tons CO2e). A key metric for environmental impact.
- **Energy_Output**: Annual energy production (in kWh). Indicates the projects contribution to energy supply.

Geospatial Dataset

- **Efisiensi_Lokasi**: Location efficiency (High, Medium, Low). Reflects suitability based on factors like solar irradiance or grid proximity.

Financial Dataset

- **Investment_Cost**: Total funds invested in the project (in billion rupiah). Indicates the projects financial scale.

Exam Questions

Below are the seven exam questions and one bonus question, each with a description, task, example output, and additional explanation to guide your approach.

Question 1: Conditional Statements (If-Else) and Arithmetic Operations

Description: The government seeks to identify PLTS projects with high CO2 reduction efficiency per unit of investment, calculated as CO2 reduction per million rupiah.

Task:

- Merge `Environmental_Dataset.xlsx` and `Financial_Dataset.xlsx` using `Project_ID`.
- For PLTS projects (`Project_ID` starts with "PLTS"), compute the ratio: `CO2_Reduction / (Investment_Cost * 1_000_000)`.
- Use `if-else` to classify the ratio as "High" (≥ 0.5 tons CO2e/million Rp) or "Low" (< 0.5).
- Display results as: "`Project_ID`: Ratio (Category)" using `f-strings`.

Example Output:

```
PLTS-NTT-001: 0.50 (High)
PLTS-JATIM-001: 0.45 (Low)
```

Additional Explanation: This question tests your ability to merge datasets, filter for specific project types, and apply conditional logic. Use dictionary operations to merge data by `Project_ID`, check for PLTS projects using string methods (e.g., `startswith()`), and compute the ratio with arithmetic operations. Ensure proper formatting with `f-strings`.

Question 2: For Loop and Lists

Description: The government needs the average CO2 reduction across PLTM projects to assess their collective environmental impact.

Task:

- Use `Environmental_Dataset.xlsx`.
- Create a list of `CO2_Reduction` values for PLTM projects (`Project_ID` starts with "PLTM").
- Use a `for` loop to calculate the total CO2 reduction and count of PLTM projects.
- Compute and display the average.

Example Output:

```
Average CO2 Reduction for PLTM Projects: 35000 tons CO2e
```

Additional Explanation: This question focuses on list creation and iteration. Filter PLTM projects using a `for` loop and string methods, append `CO2_Reduction` values to a list, and compute the average by dividing the sum by the count. Handle empty lists to avoid division by zero.

Question 3: While Loop and User Input

Description: The government requires a tool to check land status and social conflict levels by entering `Project_ID`s.

Task:

- Use `Social_Dataset.xlsx`.
- Write a program using a `while` loop to prompt for `Project_ID` until "DONE" is entered.
- For valid `Project_ID`s, display `Land_Status` and `Tingkat_Konflik`.
- For invalid `Project_ID`s, show "Project not found".

Example Output:

```
Enter Project_ID (or 'DONE' to finish): PLTS-NTT-001
PLTS-NTT-001 - Land Status: Adat, Tingkat Konflik: High
Enter Project_ID (or 'DONE' to finish): INVALID-ID
Project not found
```

```
Enter Project_ID (or 'DONE' to finish): DONE
```

Additional Explanation: This question tests user input handling and loop control. Use a dictionary to store social data, check for valid `Project_ID`s, and format output with `f-strings`. Ensure the loop terminates correctly when "DONE" is entered.

Question 4: Dictionary and Conditional Filtering

Description: The government seeks projects with high investment attractiveness and low social conflict to minimise risks.

Task:

- Merge `Economic_Dataset.xlsx` and `Social_Dataset.xlsx` using `Project_ID`.
- Create a dictionary with `Project_ID` as keys and a tuple (`Daya_Tarik_Investasi`, `Tingkat_Konflik`) as values.
- Use a `for` loop with `if` to filter projects where `Daya_Tarik_Investasi == "High"` and `Tingkat_Konflik == "Low"`.
- Display the filtered `Project_ID`s.

Example Output:

```
Projects with High Investment Attractiveness and Low Conflict:
PLTS-JATIM-001
PLTS-NTB-001
```

Additional Explanation: This question combines dictionary operations and conditional filtering. Merge datasets into a dictionary, iterate with `items()`, and apply conditions to filter projects. Ensure accurate string comparisons for categorical values.

Question 5: Functions and Arithmetic

Description: The government needs to calculate the total investment for projects with high location efficiency.

Task:

- Define a function `calculate_total_investment` that takes a list of `Project_ID`s and merged data from `Geospatial_Dataset.xlsx` and `Financial_Dataset.xlsx`.
- Use a `for` loop to sum `Investment_Cost` for projects where `Efisiensi_Lokasi == "High"`.
- Return and display the total.

Example Output:

```
Total Investment for High-Efficiency Locations: 875.73 billion Rp
```

Additional Explanation: This question tests function definition and data processing. Create a function that filters projects based on `Efisiensi_Lokasi`, sums `Investment_Cost`, and returns the result. Ensure proper handling of merged data structures.

Question 6: Modules and Error Handling

Description: The government requires a reusable tool to compute CO2 reduction efficiency with error handling.

Task:

- Create a module `green_analysis.py` with a function `compute_co2_efficiency` that takes `CO2_Reduction` and `Investment_Cost` as parameters.
- Use `try-except` to handle `ZeroDivisionError` (if `Investment_Cost` is 0), returning "Cannot compute" if an error occurs.
- Otherwise, compute and return the ratio: `CO2_Reduction / (Investment_Cost * 1_000_000)`.
- In the main script, import the module and test it on three projects.

Example Output:

```
PLTS-NTT-001: 0.50
PLTS-JATIM-001: 0.45
PLTM-PAPU-001: 0.40
```

Additional Explanation: This question focuses on modularity and error handling. Create a separate module file, implement error handling for division, and test the function with sample data. Ensure proper import and output formatting.

Question 7: Error Handling in Loops

Description: The government needs to calculate the average energy output of selected projects, handling missing data.

Task:

- Create a list of `Project_ID`s to analyse.
- Use a `for` loop with `try-except` to process `Energy_Output` from `Environmental_Dataset.xlsx`, catching `KeyError` for missing `Project_ID`s.
- Sum valid `Energy_Output` values and count valid projects.
- Compute and display the average.

Example Output:

```
Average Energy Output: 22000 kWh
```

Additional Explanation: This question tests error handling within loops. Use a `try-except` block to handle missing data, accumulate valid values, and compute the average. Ensure robust error handling to avoid crashes.

Bonus Question: Machine Learning/AI with Decision Tree

Description: The government aims to predict investment attractiveness ("High", "Medium", "Low") for new projects based on features like `GDP_Growth`, `CO2_Reduction`, and `Investment_Cost`.

Task:

- Merge `Economic_Dataset.xlsx`, `Environmental_Dataset.xlsx`, and `Financial_Dataset.xlsx`
- Use scikit-learn to build a Decision Tree Classifier with `Daya_Tarik_Investasi` as the target.
- Train the model, evaluate its accuracy, and predict the attractiveness of a new project (e.g., `GDP_Growth=5.0`, `CO2_Reduction=70000`, `Investment_Cost=150`).

Concepts and Methods: A Decision Tree Classifier splits data into branches based on feature values to classify outcomes. The process involves:

- **Data Preparation**: Merge datasets and select features (`GDP_Growth`, `CO2_Reduction`, `Investment_Cost`) and target (`Daya_Tarik_Investasi`).
- **Splitting**: Use `train_test_split` to divide data into training (80%) and testing (20%) sets.
- **Training**: Fit the Decision Tree model using the training data.
- **Evaluation**: Compute accuracy on the test set using `accuracy_score`.
- **Prediction**: Predict the class for new data.

Reference: Scikit-learn Decision Trees (https://scikit-learn.org/stable/modules/tree.html)

Example Output:

```
Model Accuracy: 0.85
Prediction for new project: High
```

Additional Explanation: This question introduces Machine Learning. Use scikit-learn to implement a Decision Tree Classifier, preprocess data by merging datasets, and evaluate model performance. Ensure proper data splitting and feature selection for accurate predictions.

Submission Guidelines

- **Deadline**: 4 July 2025, by 21:00 WIB
- **Method**: Upload your completed task to your groups GitHub repository, ensuring all team members are linked.
- **Format**: Include well-commented Python code for each question, a concise explanation of your approach, and the required output or predictions.

Additional Instructions

- **Collaboration**: Collaboration is encouraged, but each team member must be able to explain the solutions to demonstrate understanding.
- **Documentation**: Provide a brief Markdown report summarising your findings and recommendations based on the data analysis.

- **Presentation**: Be prepared to present your findings during the subsequent session, as this will contribute to your overall assessment.

Important Notes

- Ensure your code is well-structured, readable, and adheres to Python style guidelines (e.g., PEP 8).
- Test your code thoroughly to verify it produces the expected results.
- For the bonus question, libraries such as scikit-learn may be used for Machine Learning tasks.

This task provides a platform to showcase your ability to apply Python programming to real-world challenges, supporting Indonesias renewable energy priorities. We encourage collaboration with your team members and utilisation of programme resources. For any questions or clarification, please contact **Supervisor Arry Hutomo** via the communication channels for Batch 9 and 10.

We wish you every success and eagerly anticipate reviewing your innovative submissions.

Sincerely,
Andrej Karpathy
Assessors Lead Coordinator ECO Techno Leaders