Task:

Building a Sequential Model for Sketch Generation from a given class or object name

Objective and Dataset:

In this assignment, you are required to implement a sequential model in Python (using PyTorch) that generates step-by-step sketches of objects when given their class names as input. You can use the Quick, Draw! dataset (https://quickdraw.withgoogle.com/data), which contains millions of drawings across hundreds of categories, collected from people playing the game Quick, Draw!

The dataset provides stroke-based drawings where each sketch is represented as a sequence of strokes. Each stroke is a sequence of points with the pen state (down, up, or end of drawing).

You need to:

1. Select at least 10 distinct object classes from the Quick, Draw! dataset
2. Implement a model that takes a class name (e.g., "apple", "cat", "airplane") as input and generates a sequence of strokes that form a recognizable sketch of that object
3. Ensure the model can produce strokes in a step-by-step manner that mimics human drawing behaviour.

[If it produces a complete image at once you will be given zero marks.]

Network Architecture:

You are required to design and implement a sequential generative model for this task. Some possible approaches include:

- Sequence-to-sequence models with attention mechanisms
- Recurrent Neural Networks (RNNs, LSTMs, or GRUs) with conditional inputs
- Transformer-based models for sequence generation

[You are free to choose any model or a new model of your own choice.]

Your model should:

- Take a class name as input (either as a one-hot encoding or embedded representation)
- Generate a sequence of strokes that form a recognizable sketch of the specified object
- Output the strokes in a sequential manner that can be visualized step-by-step

Implementation Requirements:

1. **Data Preprocessing (10 points)**:
    - Properly load and preprocess the stroke data from the Quick, Draw! dataset
    - Implement appropriate normalization and tokenization for both the class names and stroke sequences
    - Split the dataset into training, validation, and test sets (70:15:15)
2. **Model Architecture (20 points)**:

- Design and implement a sequential generative model of your own choice.
- Justify your architecture choices in the report
- Use appropriate embedding techniques for class names
- Implement a suitable mechanism for generating sequential strokes

3. **Training and Evaluation (15 points)**:
    - Train the model with appropriate loss functions and optimization techniques
    - Implement early stopping and learning rate scheduling
    - Monitor and report training progress
4. **Visualization and Analysis (5 points)**:
    - Develop a visualization tool that shows the step-by-step generation of sketches
    - Compare generated sketches with real human-drawn examples
    - Analyze where the model succeeds and fails

    [You have to provide the real-time visualisation of your saved model during the evaluation of the assignment. Failing to do it will result in zero marks.]

BONUS (10 points):

Implement one or more of the following extensions:

1. **Interactive Refinement (5 points)**:
    - Implement a mechanism for users to provide feedback or corrections to the generated sketches
    - The model should be able to adapt its output based on this feedback
2. **Multi-object Composition (5 points)**:
    - Extend your model to generate scenes with multiple objects
    - Take a list of object names as input and compose them into a coherent scene

References and Resources:

- Quick, Draw! Dataset: https://quickdraw.withgoogle.com/data