Ainslee's RL Final Project Environment Design

This environment simulates a character-level spell correction task. Each episode begins with a real English word that has been corrupted using DeepWordBug. The agent sees the corrupted version and attempts to restore it to the correct (original) spelling using one-character substitutions. The environment is built to be compatible with OpenAI Gymnasium and can be used with discrete-action RL algorithms like PPO or DQN.

- **Observation Space**
  - `obs`: a fixed-length sequence of characters (e.g., max 10), padded as needed.
  - Characters are integer-encoded (`a=1` to `z=26`, `PAD=0`).
  - Represented as a 1D NumPy array of length `max_word_length`
- **Action Space**
  - Discrete space of size `max_word_length × 26`.
    - Each action encodes a tuple `(position, new_char)`, where:
      - `position` ∈ `[0, max_word_length - 1]`
      - `new_char` ∈ `[0, 25]`, representing `'a'` through `'z'`
    - To decode:
      ```
      position = action // 26
      char_index = action % 26
      new_char = chr(97 + char_index)  # ASCII for 'a' is 97
      ```
    - This is functionally the same as using `(position, letter)` directly, but flattened into a single integer for Gymnasium's `Discrete(n)` action space. This keeps the interface simple for standard agents that expect discrete actions.
    - I could instead define a `MultiDiscrete([max_word_length, 26])` action space if I'd rather keep the (position, char) tuple literal.
- **Step Output Format**
  - The `step(action)` method returns:
    `obs, reward, terminated, truncated, info`
    - `obs`: the new word state, integer-encoded and padded
    - `reward`: see reward structure below
    - `terminated`: `True` if the agent has restored the word exactly
    - `truncated`: `True` if max number of steps reached
    - `info`: a dict with optional metadata (maybe: edit distance or original word)
- **Reward Structure**
  - Sparse by default:
    - `+1.0` if the current word exactly matches the ground truth (i.e., the correct word **before** it was corrupted with DeepWordBug)
    - `0.0` otherwise
  - Optional: *negative shaping via normalized edit distance*:
    - The agent receives `-normalized_edit_distance(current,`

`target)`, where:
      - `normalized_edit_distance = edit_distance / max(len(current), len(target))`
      - This penalizes edits that move the agent farther from the correct answer.
      - Encourages more efficient and targeted edits.
    - This can be introduced if sparse rewards prove too difficult for learning.
- **Done Condition**
  - `terminated = True` if current word == original word (i.e., correct spelling)
  - `truncated = True` if step count $\geq$ to `max_steps`
  - The episode ends when either is `True`.