Example idea: Python Dictionaries

Node Structure:

Each node in the graph should represent a specific sub-concept or skill related to Python dictionaries. You can classify these nodes into different categories such as basic, intermediate, and advanced. These nodes will be connected based on prerequisite knowledge or logical relationships.

Example Nodes:

Basic Nodes:

- Creating a Dictionary: Knowledge of how to define a dictionary using `{}` or the `dict()` function.
- Accessing Values: Accessing values using keys with `dict[key]`.
- Checking Key Existence: Using the `in` keyword to check if a key exists in a dictionary.

Intermediate Nodes:

- Adding/Modifying Entries: How to add or modify key-value pairs in a dictionary.

- Iterating Through a Dictionary: Using loops (for key in dict:) to iterate through keys or values.
- Dictionary Methods: Familiarity with common methods like `.get()`, `.keys()`, `.values()`, `.items()`.

Advanced Nodes:

- Dictionary Comprehension: Creating dictionaries using comprehension.
- Nested Dictionaries: Using dictionaries inside dictionaries (e.g., `{key1: {subkey1: value1}}`).
- Complex Key Types: Understanding that keys must be `immutable`, allowing types like `strings` or `tuples` but not lists.

Spectrum of Understanding:

Represent a student's understanding of each node on a qualitative scale like:

- 0: No understanding
- 1: Basic familiarity
- 2: Moderate understanding
- 3: Proficient
- 4: Expert

Perhaps we can also incorporate the forget curve here. These values do not need to be `integers`. They can be `doubles` and we slap a forget curve equation on here based on the last time the student's score for this node was updated and adjust the scale accordingly? (Perhaps strech goal for this as well).

Example:

```
dictNode = Node(
    idea = "What is a dictionary",
    studentUnderstanding = 3
)
```

Connections Between Nodes:

The connections (edges) between nodes can represent prerequisite relationships or conceptual dependencies.

Example:

- `"Accessing Values"` might depend on `"What is a Dictionary"` because you need to understand the concept of a dictionary before you can access its values.
- `"Iterating Through a Dictionary"` could connect to both `"What is a Dictionary"` and `"Accessing Values"` because understanding both concepts is necessary to iterate effectively.
- Edges could also represent conceptual reinforcement. For example, understanding `"Dictionary Comprehension"` might reinforce the knowledge of `"Adding/Modifying Entries"` and `"Iterating Through a Dictionary."`

Handling Misconceptions (Perhaps strech goal??):

Incorporate nodes that represent misconceptions or common mistakes:

```
mc = Node(
    idea = "Using Lists as Keys",
    studentMis = (0 or 1)
)
```

Tracking Progress:

As students interact with PyTutor using code editor and conversations, can adjust their understanding of each node.

Example,

After successfully completing a task that involves modifying a dictionary, the student's `"Adding/Modifying Entries"` node might move from `2` (Moderate Understanding) to `3` (Proficient).

Additional Features (Strech Goal???):

- Feedback Mechanism: If a student is stuck on a concept, the graph could suggest reviewing related foundational nodes. (Using the prereq graph)