Interactive Python Learning Companion (Full Version)

Includes 5 topics, input handling, hinting, and real-time feedback. This agent helps users learn Python with adaptive lessons and interactive exercises.

Features

- Personalized topics based on learner progress
- Exercise generation with hints
- Tracks scores, attempts, and usage of hints
- Real-time feedback and scoring

Usage

Open the notebook in Google Colab and follow the instructions to interact with the learning agent.

Author

Amisha Singleton

```
from IPython.display import display, Markdown, clear_output
import random
import json

# === Learner Profile with 5 Topics ===
learner_profile = {
    "name": "AutoUser",
    "level": "beginner",
    "progress": {
        "variables": {"score": 0, "attempts": 0},
        "loops": {"score": 0, "attempts": 0},
        "functions": {"score": 0, "attempts": 0},
        "conditionals": {"score": 0, "attempts": 0},
        "lists": {"score": 0, "attempts": 0}
    },
    "history": [],
    "hints_used": 0
}


# === Reasoning Component: Choose Weakest Topic ===
def choose_topic():
    ratios = {
        topic: (data["score"] / data["attempts"] if data["attempts"] else 0)
        for topic, data in learner_profile["progress"].items()
    }
    return min(ratios, key=ratios.get)


# === Exercise Generator ===
def generate_exercise(topic):
```

```
    questions = {
        "variables": {
            "question": "What is the output of this code?\n```python\nx = 5\ny = 10\nprint(x + y)```",
            "answer": "15",
            "hint": "Add x and y."
        },
        "loops": {
            "question": "What is the output?\n```python\nfor i in range(3): print(i)```",
            "answer": "0\n1\n2",
            "hint": "Prints numbers 0 to 2."
        },
        "functions": {
            "question": "What does this return?\n```python\ndef add(a, b): return a + b\nadd(3, 4)```",
            "answer": "7",
            "hint": "Adds 3 and 4."
        },
        "conditionals": {
            "question": "What is printed?\n```python\nx = 4\nif x > 2:\n    print('Greater')\nelse:\n
            "answer": "Greater",
            "hint": "x is greater than 2."
        },
        "lists": {
            "question": "What is the output?\n```python\nnums = [1, 2, 3]\nprint(nums[1])```",
            "answer": "2",
            "hint": "Lists are zero-indexed."
        }
    }
    return questions[topic]


# === Assess User Response ===
def assess_response(user_answer, correct_answer):
    return user_answer.strip() == correct_answer.strip()


# === Update Learner Progress ===
def update_profile(topic, correct, hint_used):
    learner_profile["progress"][topic]["attempts"] += 1
    if correct:
        learner_profile["progress"][topic]["score"] += 1
    if hint_used:
        learner_profile["hints_used"] += 1


# === Run Session Automatically ===
def run_auto_session(rounds=5):
    for _ in range(rounds):
        clear_output()
        topic = choose_topic()
        exercise = generate_exercise(topic)
        learner_profile["history"].append(exercise["question"])

        display(Markdown(f"### Topic: **{topic.capitalize()}**"))
        display(Markdown(exercise["question"]))
        display(Markdown(f"💡 Hint: {exercise['hint']}"))

        # Simulate correct answer (for demo)
        user_answer = exercise["answer"]
        correct = assess_response(user_answer, exercise["answer"])
```

```
        update_profile(topic, correct, hint_used=True)

        print("✅ Simulated Answer Correct")
        print("--- Progress Summary ---")
        for t, data in learner_profile["progress"].items():
            print(f"{t.title()}: {data['score']}/{data['attempts']} correct")


# Run session
run_auto_session(rounds=5)
```

Topic: Lists

What is the output? `python nums = [1, 2, 3] print(nums[1])`

💡 Hint: Lists are zero-indexed.

```
✅ Simulated Answer Correct
--- Progress Summary ---
Variables: 1/1 correct
Loops: 1/1 correct
Functions: 1/1 correct
Conditionals: 1/1 correct
Lists: 1/1 correct
```