Name:Sanika Deshmukh

Domain:Python

Code:

```
import json
import os
from dataclasses import dataclass
from typing import List

DATA_FILE = "quiz_data.json"

@dataclass
class Question:
    prompt: str
    options: List[str]
    answer_index: int

    @classmethod
    def from_dict(cls, data: dict) -> "Question":
        return cls(
            prompt=data["prompt"],
            options=data["options"],
            answer_index=int(data["answer_index"]),
        )

    def to_dict(self) -> dict:
        return {
            "prompt": self.prompt,
            "options": self.options,
            "answer_index": self.answer_index,
        }
```

```
def ensure_sample_data() -> None:
    """Create sample quiz data if none exists."""
    if os.path.exists(DATA_FILE):
        return
    sample = [
        Question(
            prompt="What is the capital of France?",
            options=["Berlin", "Paris", "Rome", "Madrid"],
            answer_index=1,
        ),
        Question(
            prompt="Which planet is known as the Red Planet?",
            options=["Venus", "Mars", "Jupiter", "Mercury"],
            answer_index=1,
        ),
        Question(
            prompt="What is the largest ocean on Earth?",
            options=["Atlantic", "Indian", "Arctic", "Pacific"],
            answer_index=3,
        ),
    ]
    save_questions(sample)


def load_questions() -> List[Question]:
    ensure_sample_data()
    with open(DATA_FILE, "r", encoding="utf-8") as f:
        data = json.load(f)
    return [Question.from_dict(item) for item in data]


def save_questions(questions: List[Question]) -> None:
    with open(DATA_FILE, "w", encoding="utf-8") as f:
```

```
        json.dump([q.to_dict() for q in questions], f, indent=2)


def display_question(q: Question, number: int, total: int) -> int:
    print(f"\nQuestion {number}/{total}")
    print(q.prompt)
    for idx, option in enumerate(q.options, start=1):
        print(f"  {idx}) {option}")

    while True:
        choice = input("Your answer (number): ").strip()
        if not choice.isdigit():
            print("Please enter a valid number.")
            continue
        ans_idx = int(choice) - 1
        if 0 <= ans_idx < len(q.options):
            return ans_idx
        print("Choice out of range. Try again.")


def run_quiz(questions: List[Question]) -> None:
    if not questions:
        print("No questions available. Add some first.")
        return

    correct = 0
    total = len(questions)
    for i, q in enumerate(questions, start=1):
        ans = display_question(q, i, total)
        if ans == q.answer_index:
            correct += 1
            print("Correct!")
        else:
```

```
            print(f"Wrong. Correct answer: {q.options[q.answer_index]}")
        print(f"Progress: {correct}/{i} correct ({(correct/i)*100:.1f}%).")

    final_score = (correct / total) * 100
    print("\nQuiz complete!")
    print(f"Final score: {correct}/{total} ({final_score:.1f}%).")

def add_question() -> None:
    prompt = input("Enter the question: ").strip()
    if not prompt:
        print("Question cannot be empty.")
        return

    options: List[str] = []
    while len(options) < 2:
        option = input(f"Enter option {len(options)+1}: ").strip()
        if option:
            options.append(option)
        else:
            print("Option cannot be empty.")
    while True:
        more = input("Add another option? [y/N]: ").strip().lower()
        if more == "y":
            option = input(f"Enter option {len(options)+1}: ").strip()
            if option:
                options.append(option)
            else:
                print("Option cannot be empty.")
        else:
            break
```

```
    print("Options:")
    for idx, opt in enumerate(options, start=1):
        print(f"  {idx}) {opt}")

    while True:
        correct_input = input("Enter the number of the correct option: ").strip()
        if correct_input.isdigit():
            correct_idx = int(correct_input) - 1
            if 0 <= correct_idx < len(options):
                break
        print("Invalid choice. Please enter a valid option number.")

    questions = load_questions()
    questions.append(Question(prompt=prompt, options=options, answer_index=correct_idx))
    save_questions(questions)
    print("Question added.")

def list_questions() -> None:
    questions = load_questions()
    if not questions:
        print("No questions available.")
        return
    for idx, q in enumerate(questions, start=1):
        print(f"{idx}. {q.prompt} (answers: {len(q.options)} options)")

def main() -> None:
    ensure_sample_data()
    actions = {
        "1": ("Take quiz", lambda: run_quiz(load_questions())),
        "2": ("Add question", add_question),
        "3": ("List questions", list_questions),
```

```
        "0": ("Exit", None),
    }

    while True:
        print(
            "\nQuiz Game\n"
            "1) Take quiz\n"
            "2) Add question\n"
            "3) List questions\n"
            "0) Exit\n"
        )
        choice = input("Select: ").strip()
        action = actions.get(choice)
        if not action:
            print("Invalid choice.")
            continue
        if choice == "0":
            print("Goodbye.")
            break
        _, fn = action
        if fn:
            fn()

if __name__ == "__main__":
    main()
```

Output:

```
Quiz Game
1) Take quiz
2) Add question
3) List questions
0) Exit

Select: 1

Question 1/3
What is the capital of France?
  1) Berlin
  2) Paris
  3) Rome
  4) Madrid
Your answer (number): 2
Correct!
Progress: 1/1 correct (100.0%).

Question 2/3
Which planet is known as the Red Planet?
  1) Venus
  2) Mars
  3) Jupiter
  4) Mercury
Your answer (number): 2
Correct!
Progress: 2/2 correct (100.0%).

Question 3/3
What is the largest ocean on Earth?
  1) Atlantic
  2) Indian
  3) Arctic
  4) Pacific
Your answer (number): 4
Correct!
```