Abstraction in C# is a way to simplify complex systems by hiding unnecessary implementation details and exposing only the essential features to the user. This helps focus on "what" an object does rather than "how" it works, making the code easier to maintain and extend. For example, in the following code, the GetQuestion method in the RandomQuiz class retrieves a random question from a list without revealing how the list is

public abstract class Quiz

{

protected List<string> questions = new List<string> { "What is C#?", "Define abstraction.", "What is polymorphism?" };

public abstract string GetQuestion();

}

public class RandomQuiz : Quiz

{

public override string GetQuestion()

{

Random random = new Random();

int \_index = random.Next(questions.Count);

string \_question = questions[\_index];

questions.RemoveAt(\_index);

return \_question;

}

}

This demonstrates how abstraction improves maintainability and allows flexibility in implementing different ways to fetch questions without changing the overall design.