Exercise 1: The Basic "Fixed Value"

Goal: Understand the syntax and restrictions of const.

Create a class named PhysicsConstants. Inside, define a constant for the speed of light (C approximate 299,792,458 m/s) and a constant for the name of the lab "CERN".

- **Challenge:** Try to change the speed of light inside a method and observe the compiler error.
- **Key Concept:** const is evaluated at **compile-time**.

Exercise 2: The Runtime Setup

Goal: Use readonly for values that aren't known until the program starts.

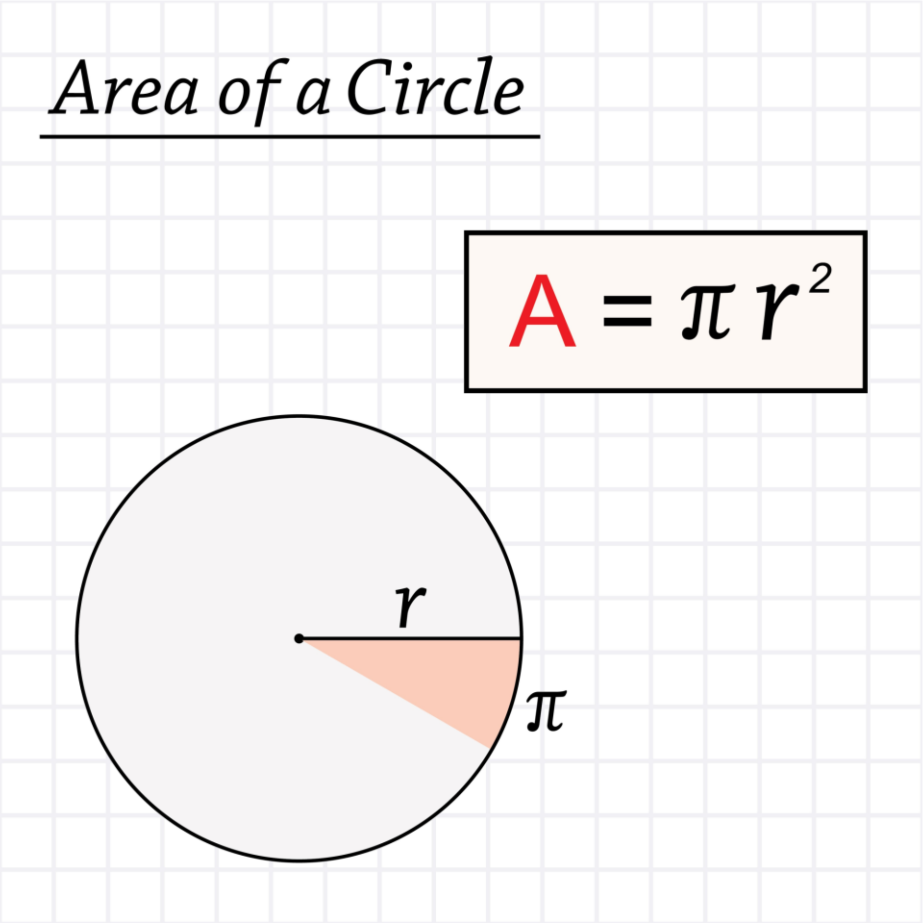
Create a class called ReportGenerator. It should have a readonly string field named _reportId.

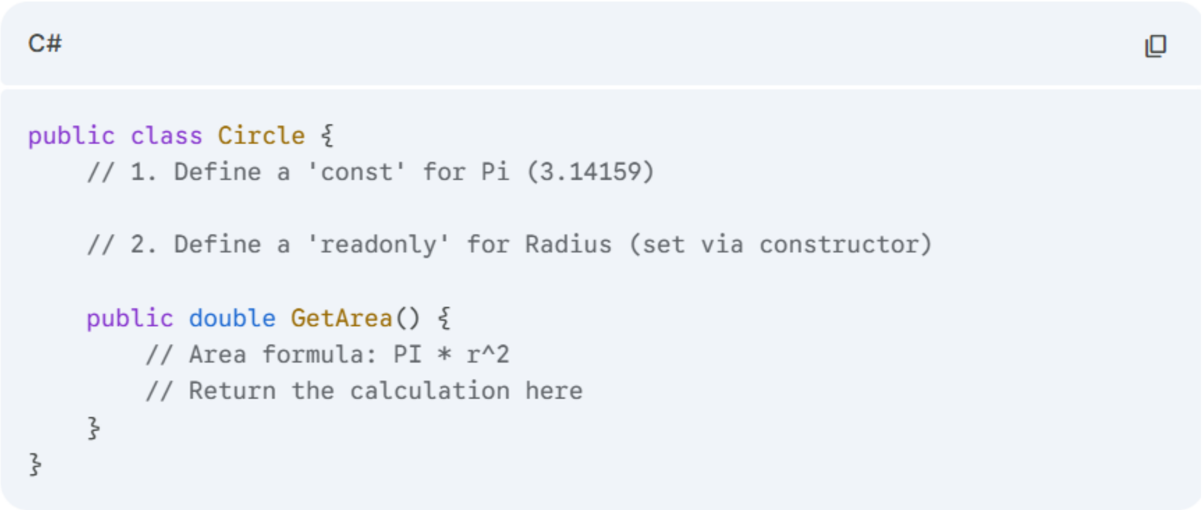
- **Task:** Initialize _reportId inside the constructor using Guid.NewGuid().ToString().
- **Challenge:** Why can’t this be a const? (Hint: Guid.NewGuid() happens while the app is running, not when it's being compiled).

Exercise 3: Const vs. Readonly Comparison

Goal: See how they interact within the same class.

Fill in the blanks for a Circle class:





```
public class Circle {
    // 1. Define a 'const' for Pi (3.14159)

    // 2. Define a 'readonly' for Radius (set via constructor)

    public double GetArea() {
        // Area formula: PI * r^2
        // Return the calculation here
    }
}
```

Exercise 4: The "Static Readonly" Pattern

Goal: Learn how to handle complex objects that shouldn't change.

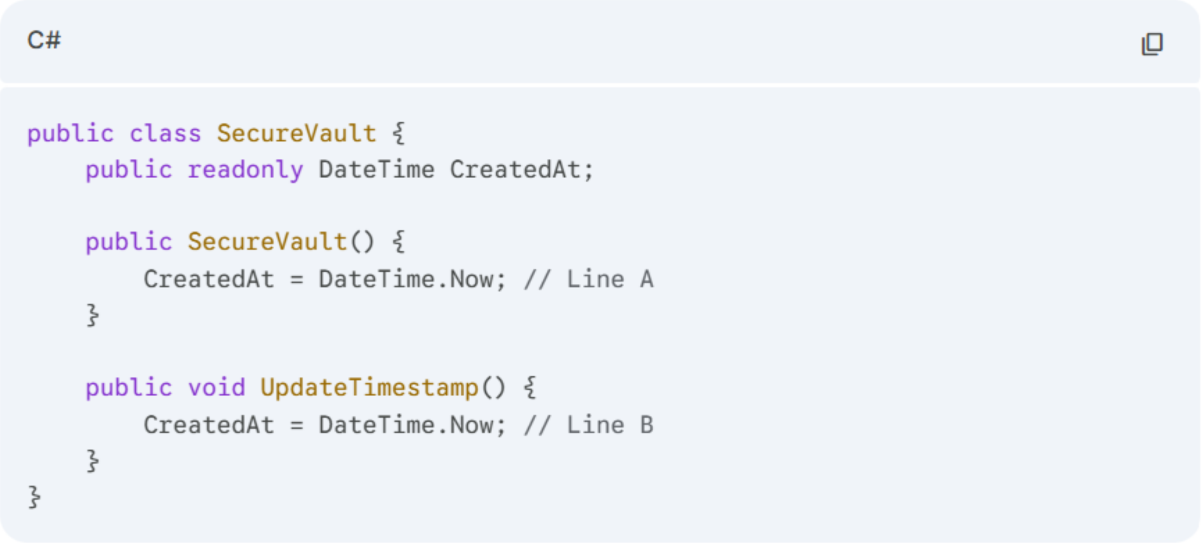
You cannot make a List or a custom User object a const because they are reference types (except for strings).

- **Task:** Create a class AppConfig. Define a public static readonly array of strings containing three "Admin" usernames.
- **Reflect:** Why is static readonly often used as a more flexible alternative to const?

Exercise 5: Debugging the Assignment

Goal: Identify where readonly protection begins and ends.

Look at the following code and determine which line will cause a compiler error and why:



C#

```
public class SecureVault {
    public readonly DateTime CreatedAt;

    public SecureVault() {
        CreatedAt = DateTime.Now; // Line A
    }

    public void UpdateTimestamp() {
        CreatedAt = DateTime.Now; // Line B
    }
}
```