Game Character System

**Introduction**

In this assignment, you will implement a simple game character system that demonstrates object-oriented programming concepts such as inheritance, method overriding, method hiding, and constructor chaining.

**Objectives**

* Demonstrate understanding of inheritance and polymorphism in C#.
* Practice using constructors to chain initialization of base classes.
* Implement method overriding and hiding to modify behavior of derived classes.

**Requirements**

* Create a base class called **Character** that has the following properties and methods:
  + **string name**
  + **int health**
  + **int attack**
  + **int defense**
  + **void Attack(Character target)**
  + **void Defend()**
* The **Attack** method should take a **Character** object as a parameter and calculate the amount of damage that the current character would do to the target character based on their attack and defense values. If the damage is greater than 0, it should reduce the target's health by that amount and print a message to the console. If the damage is less than or equal to 0, it should print a different message to the console.
* The **Defend** method should simply print a message to the console indicating that the character is defending.

You will then create three derived classes called Warrior, Mage and Dragon that inherit from Character and have the following additional methods:

**Warrior**

* Charge(Character target): a method to charge at a target
* Override the Defend method to add 5 to the character's defense value when called.

**Mage**

* CastSpell(Character target): a method to cast a spell at a target
* Seal the Defend method so that it cannot be further overridden in any derived classes.

**Dragon**

* BreatheFire(Character target): a method for the dragon to breathe fire at a target
* Override the Attack method to double the attack value when called.

You will also create a class called Game that has a main method to test your character system. In the main method, you will create one instance of each character type and have them attack each other in a loop until one of them runs out of health. You can have each character take turns attacking and defending. You can demonstrate few calls to Attack(), Defend(), BreatheFire() methods for demonstration.

**Tips**

* Use the **base** keyword to call the constructor of the base class from a derived class.
* Use the **override** keyword to modify the behavior of a method in a derived class.
* Use the **sealed** keyword to prevent further overriding of a method in a derived class.

Sample output:

1. Arthur is a warrior.
2. Merlin is a mage.
3. Smaug is a dragon.
5. Arthur attacks Merlin for 15 damage!
6. Merlin cannot defend!
7. Smaug breathes fire on Arthur for 110 damage!
8. Smaug defends and gains 10 defense.
9. Arthur defends and gains 5 defense.
10. --- Battle between Arthur and Merlin ---
11. Merlin wins!
12. --- Battle between Smaug and Arthur ---
13. Smaug wins!
14. --- Battle between Merlin and Smaug ---
15. Merlin's attack has no effect on Smaug.
16. Smaug attacks Merlin for 25 damage!
17. Merlin's attack has no effect on Smaug.
18. Smaug attacks Merlin for 25 damage!
19. Merlin's attack has no effect on Smaug.
20. Smaug attacks Merlin for 25 damage!
21. Smaug wins!