Kole S. Bauer

6/29/2024

CSE-210

W10: Articulate  
  
 Polymorphism is a hugely important concept in object-oriented programming that allows objects to be treated as instances of their parent class rather than their actual class. The term polymorphism means "many shapes" and it refers to the ability of different classes to be treated as instances of the same class through inheritance. Polymorphism is achieved in two main ways: method overriding and method overloading. An application of polymorphism can be seen in the design of a goal management system where different types of goals (simple goals, eternal goals, checklist goals) can be handled through a common interface. This allows for a cleaner and more organized code structure, as each specific goal type can have its own implementation of methods while being treated as a general goal type. In the program we wrote, we used polymorphism to manage different types of goals. Here is a code example demonstrating polymorphism through method overriding:  
  
```

// Base class

public abstract class Goal

{

protected string \_name;

protected int \_points;

protected bool \_completed;

public Goal(string name, int points)

{

\_name = name;

\_points = points;

\_completed = false;

}

public abstract int RecordEvent();

public abstract string ToDataString();

public abstract bool IsComplete();

public abstract string Display();

}

// Derived class for SimpleGoal

public class SimpleGoal : Goal

{

public SimpleGoal(string name, int points) : base(name, points) { }

public override int RecordEvent()

{

\_completed = true;

return \_points;

}

public override string ToDataString()

{

return $"SimpleGoal:{\_name},{\_points},{\_completed}";

}

public override bool IsComplete() => \_completed;

public override string Display()

{

return \_completed ? $"[X] {\_name} - {\_points} points" : $"[ ] {\_name} - {\_points} points";

}

}

// Derived class for EternalGoal

public class EternalGoal : Goal

{

public EternalGoal(string name, int points) : base(name, points) { }

public override int RecordEvent()

{

return \_points;

}

public override string ToDataString()

{

return $"EternalGoal:{\_name},{\_points}";

}

public override bool IsComplete() => false;

public override string Display()

{

return $"[ ] {\_name} - {\_points} points (Eternal)";

}

}

// Derived class for ChecklistGoal

public class ChecklistGoal : Goal

{

private int \_targetCount;

private int \_bonusPoints;

private int \_currentCount;

public ChecklistGoal(string name, int points, int targetCount, int bonusPoints)

: base(name, points)

{

\_targetCount = targetCount;

\_bonusPoints = bonusPoints;

\_currentCount = 0;

}

public override int RecordEvent()

{

\_currentCount++;

if (\_currentCount >= \_targetCount)

{

\_completed = true;

return \_points + \_bonusPoints;

}

return \_points;

}

public override string ToDataString()

{

return $"ChecklistGoal:{\_name},{\_points},{\_targetCount},{\_bonusPoints},{\_currentCount},{\_completed}";

}

public override bool IsComplete() => \_completed;

public override string Display()

{

return \_completed

? $"[X] {\_name} - {\_points} points, Completed {\_currentCount}/{\_targetCount} times, Bonus: {\_bonusPoints} points"

: $"[ ] {\_name} - {\_points} points, Completed {\_currentCount}/{\_targetCount} times";

}

}  
```