Classes and Objects in C#

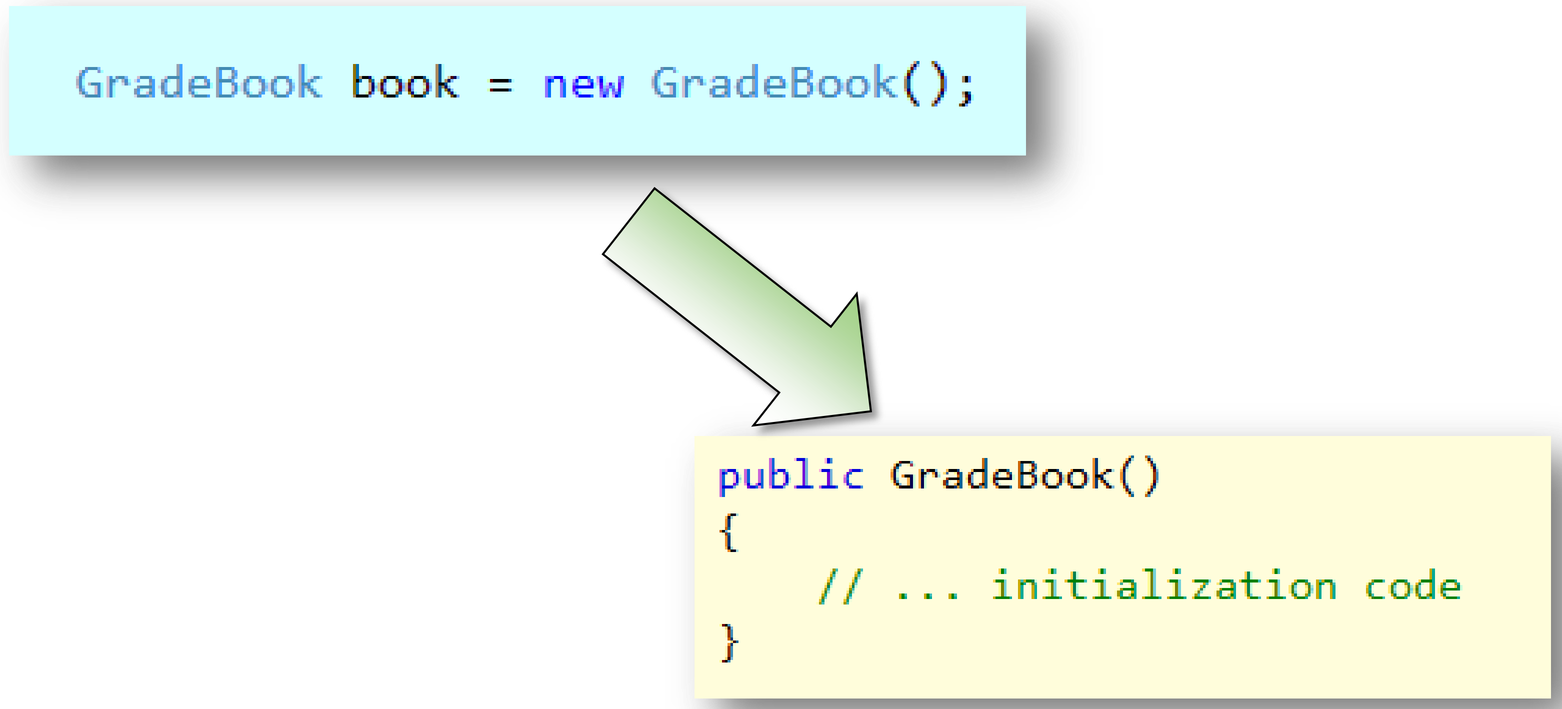
K. Scott Allen
odetocode.com
@OdeToCode

We need an electronic grade book to read the scores of an individual student and then compute some simple statistics from the scores.

The grades are entered as floating point numbers from 0 to 100, and the statistics should show us the highest grade, the lowest grade, and the average grade.

Constructors

- Special methods used to initialize objects

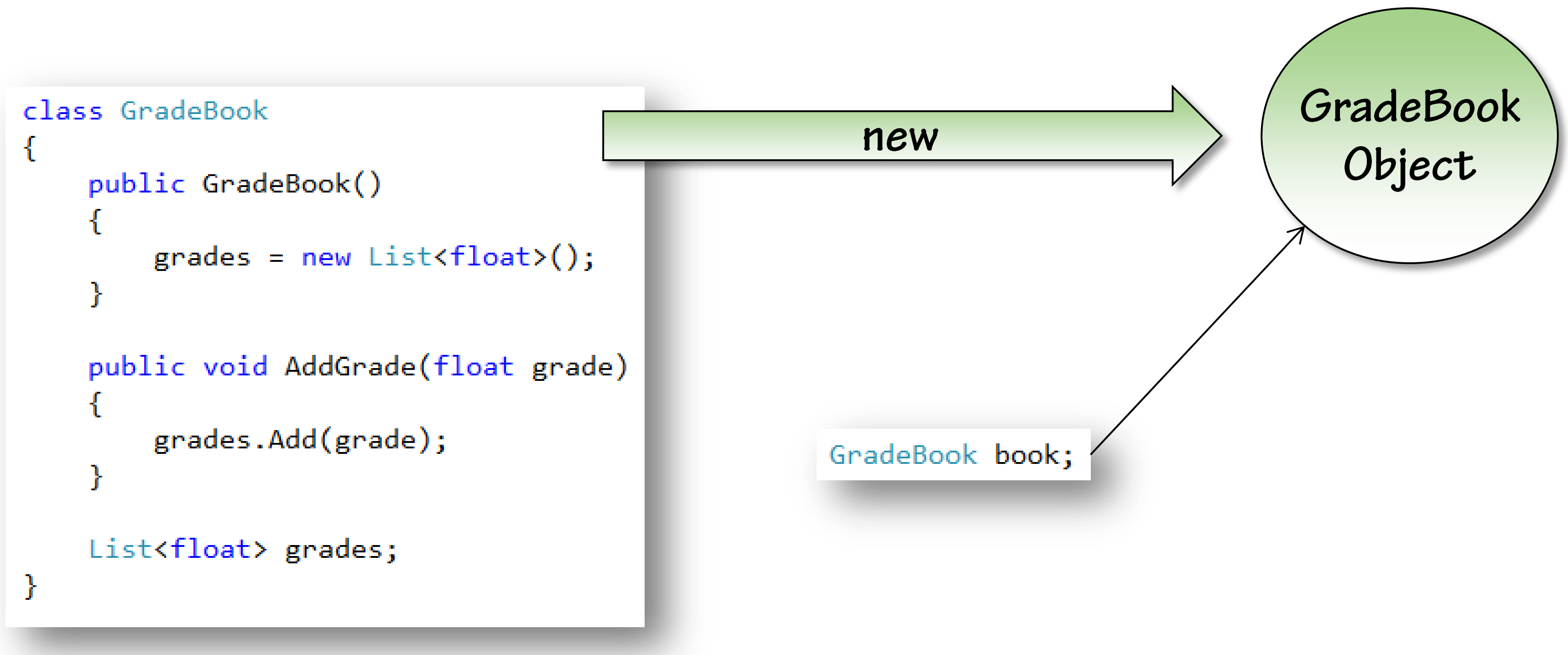


```
GradeBook book = new GradeBook();
```

```
public GradeBook()
{
    // ... initialization code
}
```

Classes Versus Variables

- A class is a blueprint for creating objects
- A class can also be used to type a variable
  - A variable can refer to any object of the same type

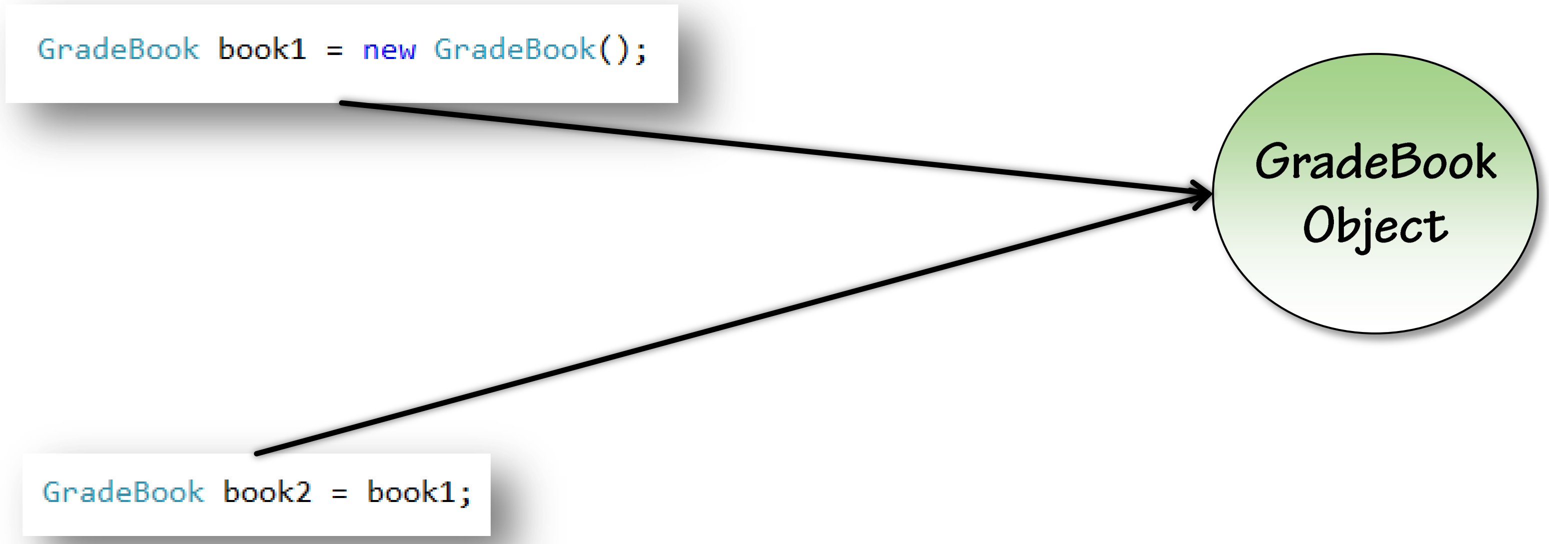


```
class GradeBook
{
    public GradeBook()
    {
        grades = new List<float>();
    }

    public void AddGrade(float grade)
    {
        grades.Add(grade);
    }

    List<float> grades;
}
```

```
GradeBook book;
```

Reference Types

- **Classes are reference types**
- **Variables point to objects**
  - They don't "hold" the objects in their storage location



```
GradeBook book1 = new GradeBook();
```

```
GradeBook book2 = book1;
```

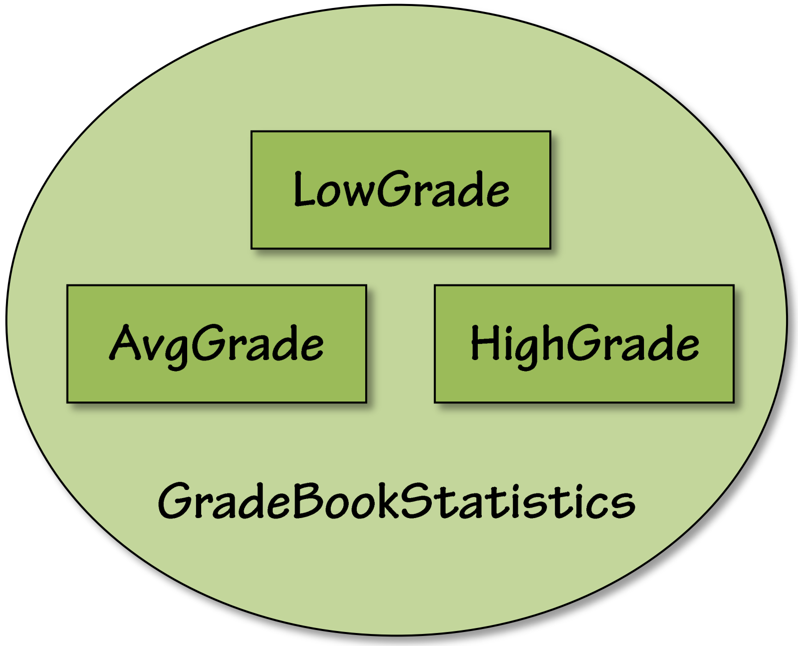
Classes

- **A class definition creates a new type**
  - Use the type for variables and arguments
- **Use a class to create objects**
  - Invoke methods and save state in objects

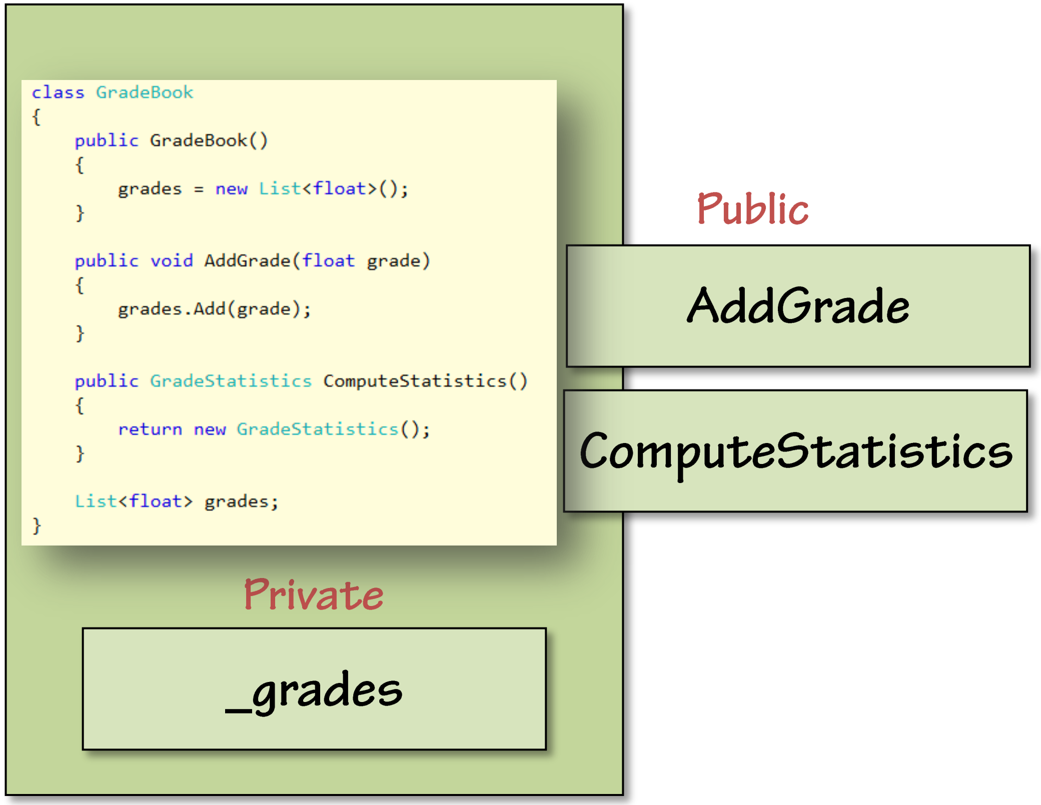
Object Oriented Programming

- Objects are nouns
- Methods are verbs
- Objects encapsulate functionality

Encapsulation



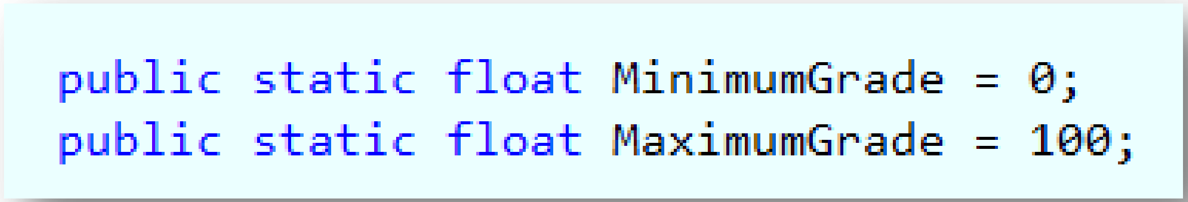
Access Modifiers



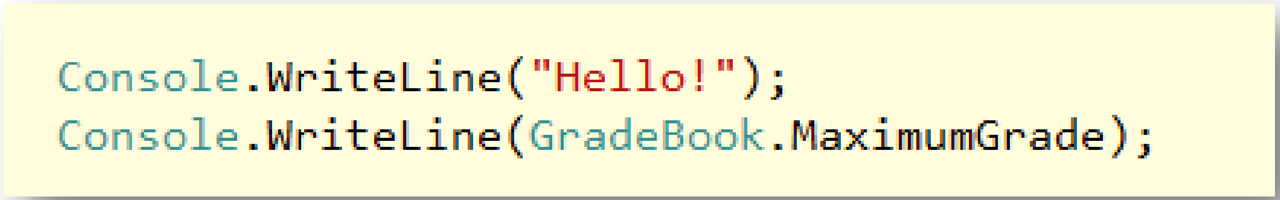
```
class GradeBook
{
    public GradeBook()
    {
        grades = new List<float>();
    }

    public void AddGrade(float grade)
    {
        grades.Add(grade);
    }

    public GradeStatistics ComputeStatistics()
    {
        return new GradeStatistics();
    }

    List<float> grades;
}
```

Statics

- Use static members of a class without creating an instance

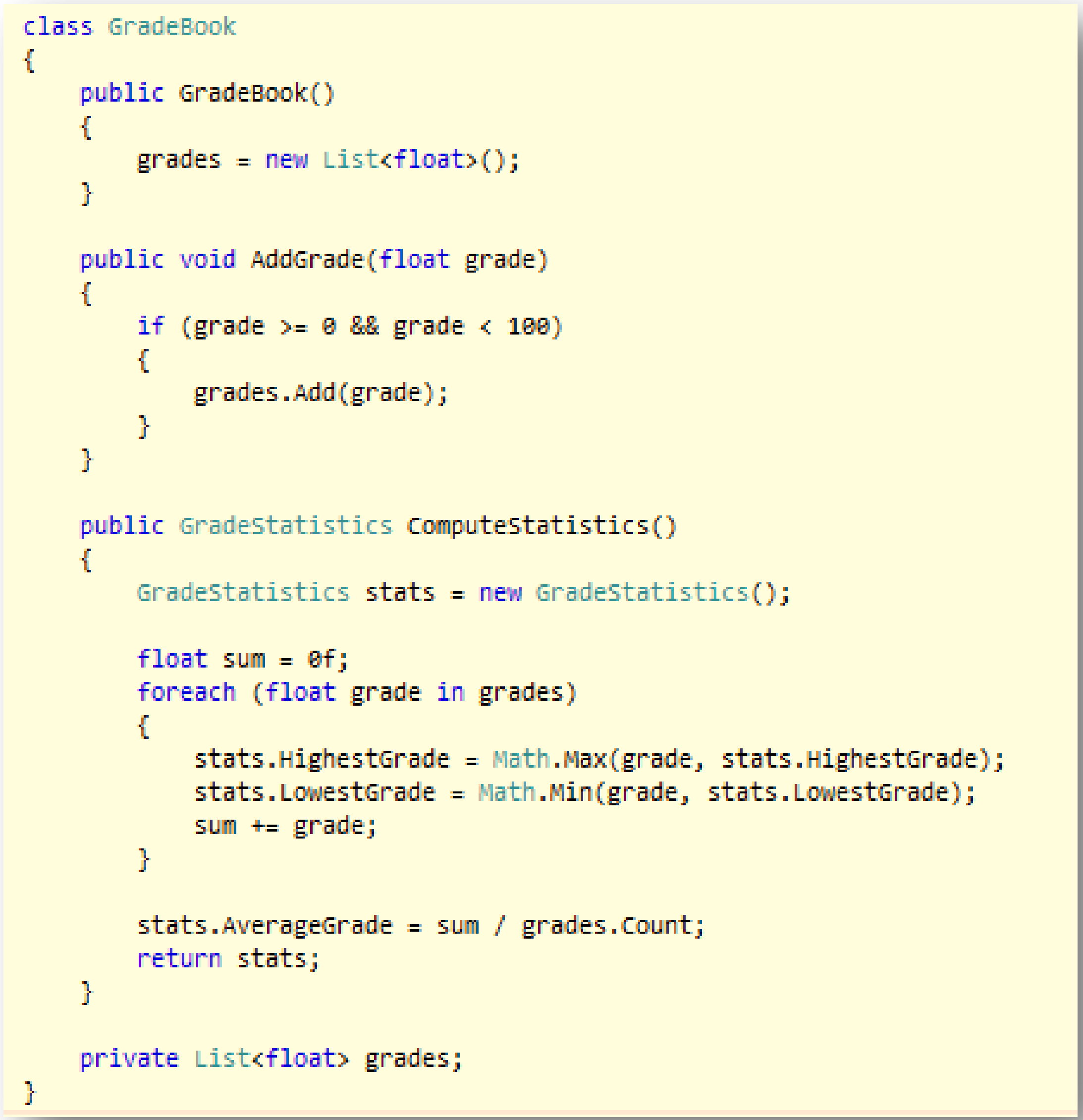


```
public static float MinimumGrade = 0;
public static float MaximumGrade = 100;
```



```
Console.WriteLine("Hello!");
Console.WriteLine(GradeBook.MaximumGrade);
```

Summary



```
class GradeBook
{
    public GradeBook()
    {
        grades = new List<float>();
    }

    public void AddGrade(float grade)
    {
        if (grade >= 0 && grade < 100)
        {
            grades.Add(grade);
        }
    }

    public GradeStatistics ComputeStatistics()
    {
        GradeStatistics stats = new GradeStatistics();

        float sum = 0f;
        foreach (float grade in grades)
        {
            stats.HighestGrade = Math.Max(grade, stats.HighestGrade);
            stats.LowestGrade = Math.Min(grade, stats.LowestGrade);
            sum += grade;
        }

        stats.AverageGrade = sum / grades.Count;
        return stats;
    }

    private List<float> grades;
}
```