OOP Design Principles

Lecture 10

Single-rooted hierarchy

- Override equals() and toString() to make your obects behave in a predictable way

Static keyword

- Class data, class method – exists for a class as a whole

Access control

- private
- public
- Package access (no keyword)
- protected

OOP benefits

- Flexibility and maintainability
- Shapes example

UML diagrams

- Composition
- Association
- Aggregation
- Inheritance
- Interfaces

OOP concepts

- Abstraction
- Encapsulation
- Composition
- Inheritance
- Polymorphism

OOP design principles

- Program to an interface, not an implementation
- Encapsulate what varies and pull it away from what stays the same
- Make your classes open for extension, but closed for modification
- Favor composition over inheritance

Design pattern 1: Strategy pattern

- Define a family of algorithms, encapsulates them, and makes them interchangeable by using a common interface
- Strategy lets the algorithm vary independently from clients that use it

- Examples and code in the next Lecture