Strategy Design Pattern

CSCI 2300

Announcements

- Homework 4 is due on Monday
- Turn in labs 9-15 by the end of today

Design Patterns

- Behavioral
  - Iterator
  - Model View Controller
  - Observer
  - **Strategy**
- Creational
- Structural

Strategy Design Pattern

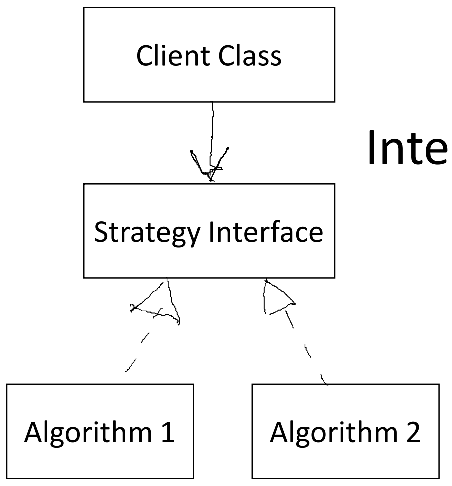
- A family of algorithms
- Algorithm behavior needs to be selected dynamically (at run time)
- Example family of algorithms: sorting algorithms
  - Bubble sort
  - Selection sort
  - Merge sort
  - Insertion sort
- Best choice of algorithm may depend on the data:
  - If data is in "almost sorted" order, insertion sort is very fast

Example Code Sketch

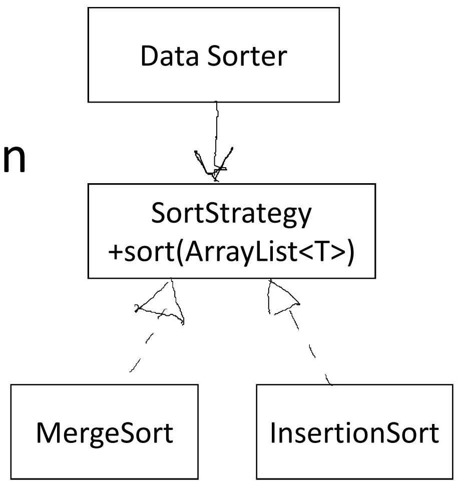
```
ArrayList<Integer> data = new ArrayList<Integer>();
…
// if data is in almost sorted order
     sortStrategy = new InsertionSort<Integer>();
// else
     sortStrategy = new MergeSort<Integer>();
DataSorter.sort(data, sortStrategy);
```

Interface Solution

General Class Diagram

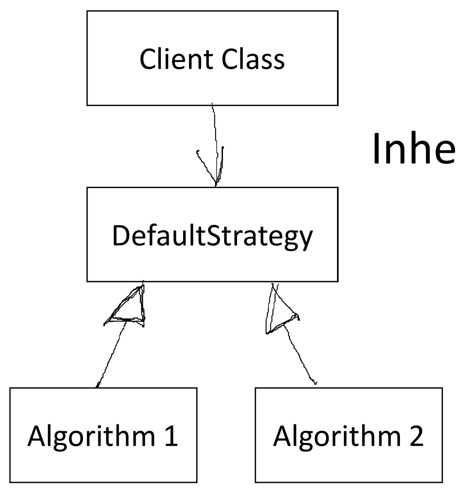


Sort Example

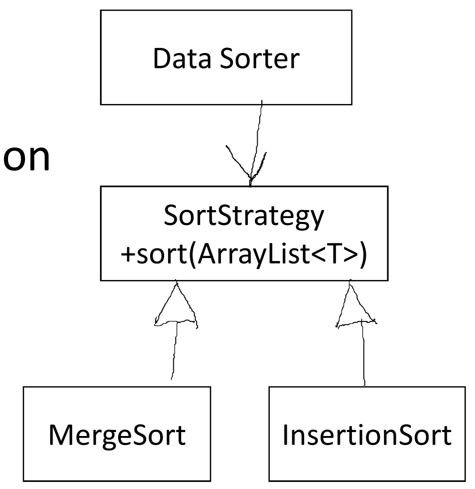


Inheritance Solution

General Class Diagram

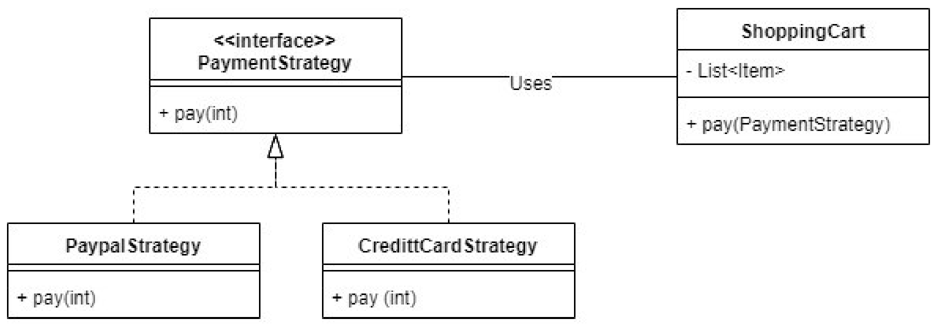


Sort Example



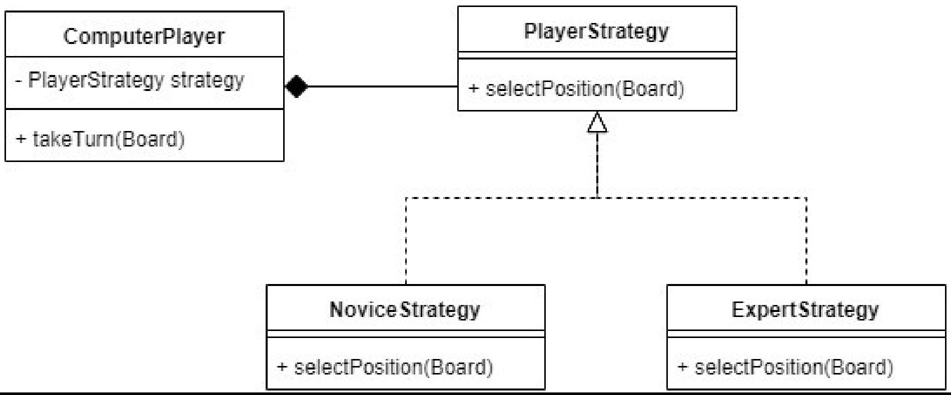
Strategy Pattern: Shopping Cart example

A checkout system allows user to select a form of payment



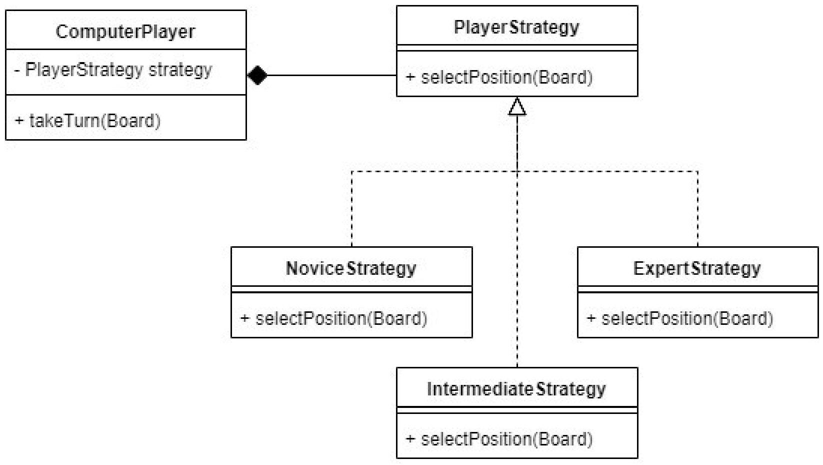
Strategy Pattern: Tic Tac Toe

- Extend Tic Tac Toe game to play against computer player
- Computer Player supports different "levels": Expert and Novice



What if we need to add a new level – intermediate?

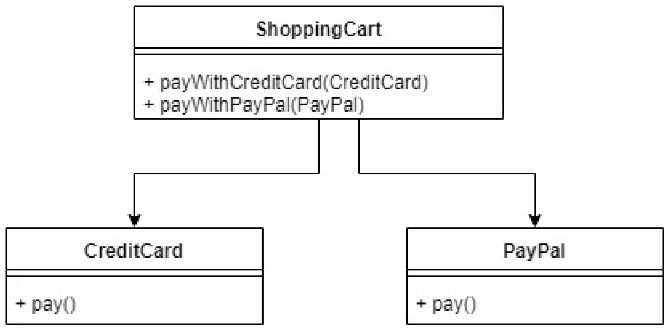
- Solution: Create an IntermediateStrategy class



Open Closed Design Principle (OCP)

- In Observer and Strategy patterns we were able to:
  - Extend existing behavior (add new functionality) without
  - Modifying existing code
- **Open** for Extension **Closed** for Modification
  - Add new functionality without changing existing code

Does this design violate Open Closed Principle?



A. This design violates OCP because `pay()` method is duplicated in `CreditCard` and `PayPal`
B. This design violates OCP because we'll need to change `ShoppingCart` if we add another payment method.
C. This design does not violate OCP: if we add another payment method, we can create a sub-class of `ShoppingCart` and add the necessary function there.
D. None of the above are true

OCP can be achieved by

- Inheritance
- Interfaces
- Design Patterns (they use inheritance and interfaces)

You are designing a “sliding puzzle” game. You need to be able to look at each square on the game and decide if it is in the correct order. Which design pattern is the best fit for this situation?

A. Iterator

B. Observer

C. Model View Controller

D. Strategy

You are implementing a game where a character moves on the screen. The user uses arrow keys to move the character. The screen needs to be updated appropriately, each time the user moves the arrow keys. What design pattern is the best fit for this situation?

A. Iterator

B. Observer

C. Model View Controller

D. Strategy