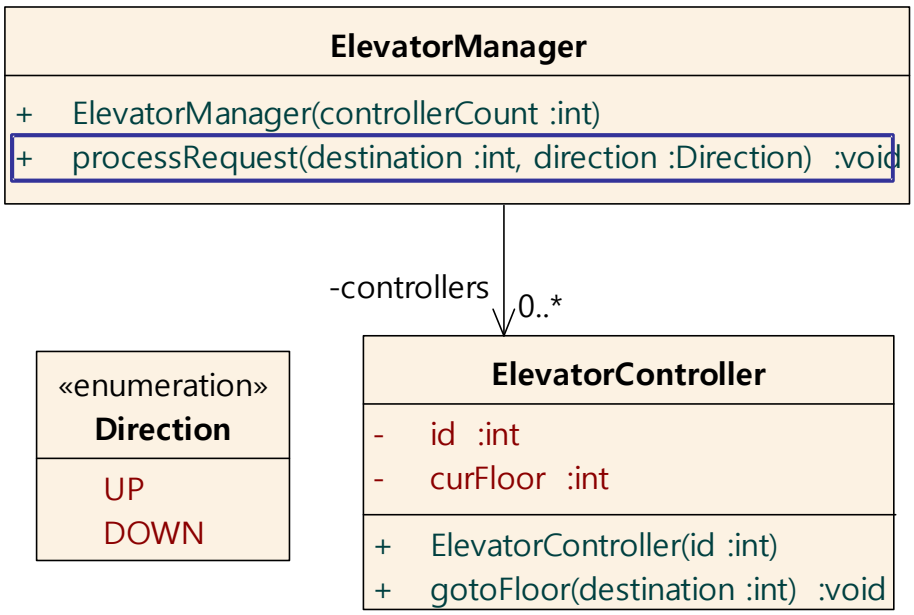
Strategy Pattern

PRACTICE – ELEVATOR SCHEDULER

Elevator Scheduling

- ElevatorManager need to select an elevator based on request.



Initial Design - ElevatorController

```
public class ElevatorController {
  private int id ;
  private int curFloor ;
  public ElevatorController(int id) {
    this.id = id ;
    curFloor = 1 ;
  }
  public void gotoFloor(int destination) {
    System.out.print("Elevator [" + id + "] Floor: " + curFloor) ;
    curFloor = destination ;
    System.out.println(" ==> " + curFloor) ;
  }
}
```

Initial Design – ElevatorManager

```
public class ElevatorManager {
  private List<ElevatorController> controllers ;
  public ElevatorManager(int controllerCount) {
    controllers = new ArrayList<ElevatorController>(controllerCount) ;
    for ( int i = 0 ; i < controllerCount ; i ++ ) {
      ElevatorController controller = new ElevatorController(i+1) ;
      controllers.add(controller) ;
    }
  }
  private void processRequest(int destination, Direction direction) {
    int selectedElevator = 0 ;

    // select an elevator to maximize throughput of the system

    controllers.get(selectedElevator).gotoFloor(destination) ;
  }
}
```


Violation of OOD Principle

Principle	Codes(class/method) that violate the principle
SRP	
OCP	
LSP	
ISP	
DIP	

Problem – Source Code

Solution – Strategy Pattern

- Declares an ElevatorScheduler interface for different

Source Code: Strategy

Source Code: Concrete Strategies

Source Code: Context

Source Code: Client