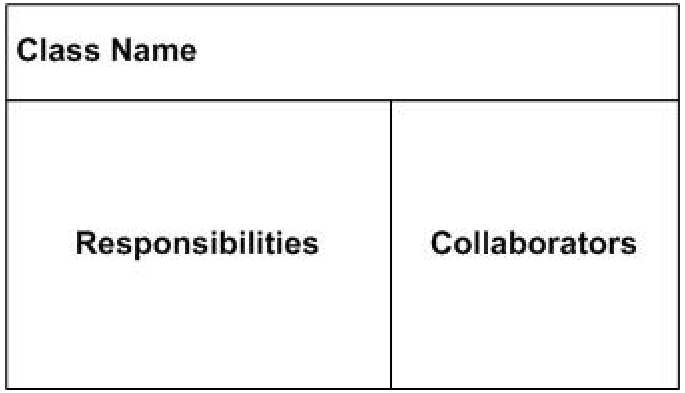
CSc 133 Lecture Notes

Assignment 1 Discussion

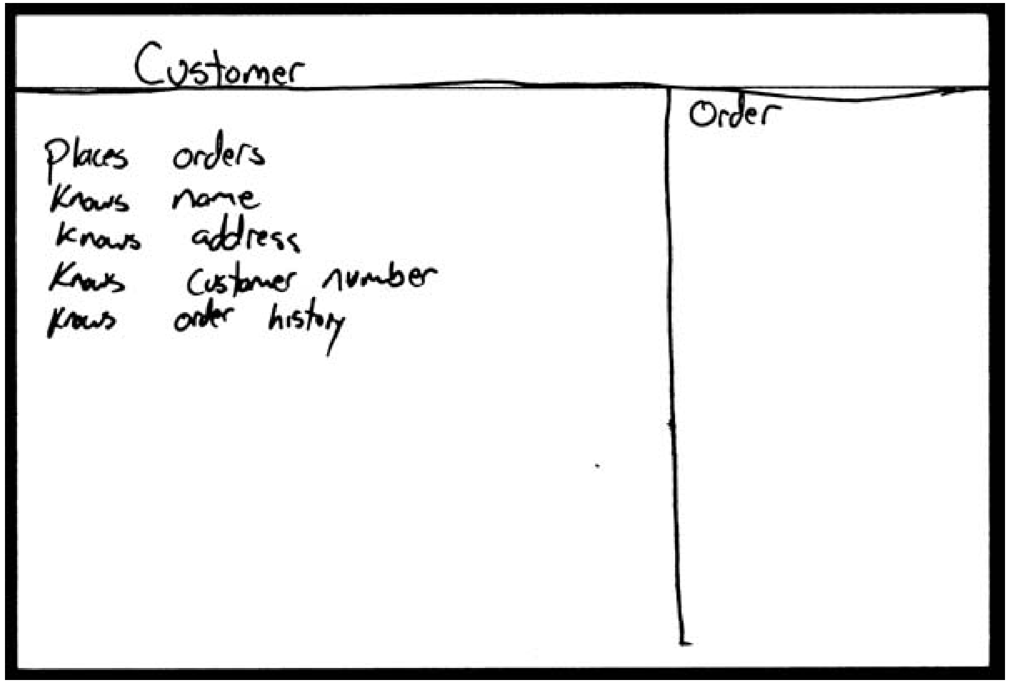
Key Points

- Your assignment 1 is a specification.
- Read the assignment MANY times.
  - See appendix at the end (please not skip it)
- Analysis
  - Identify Suitable Classes
    - Use hints
    - Use CRC form: Class Responsibility Collaboration (See appendix)
  - Check your work again your requirements

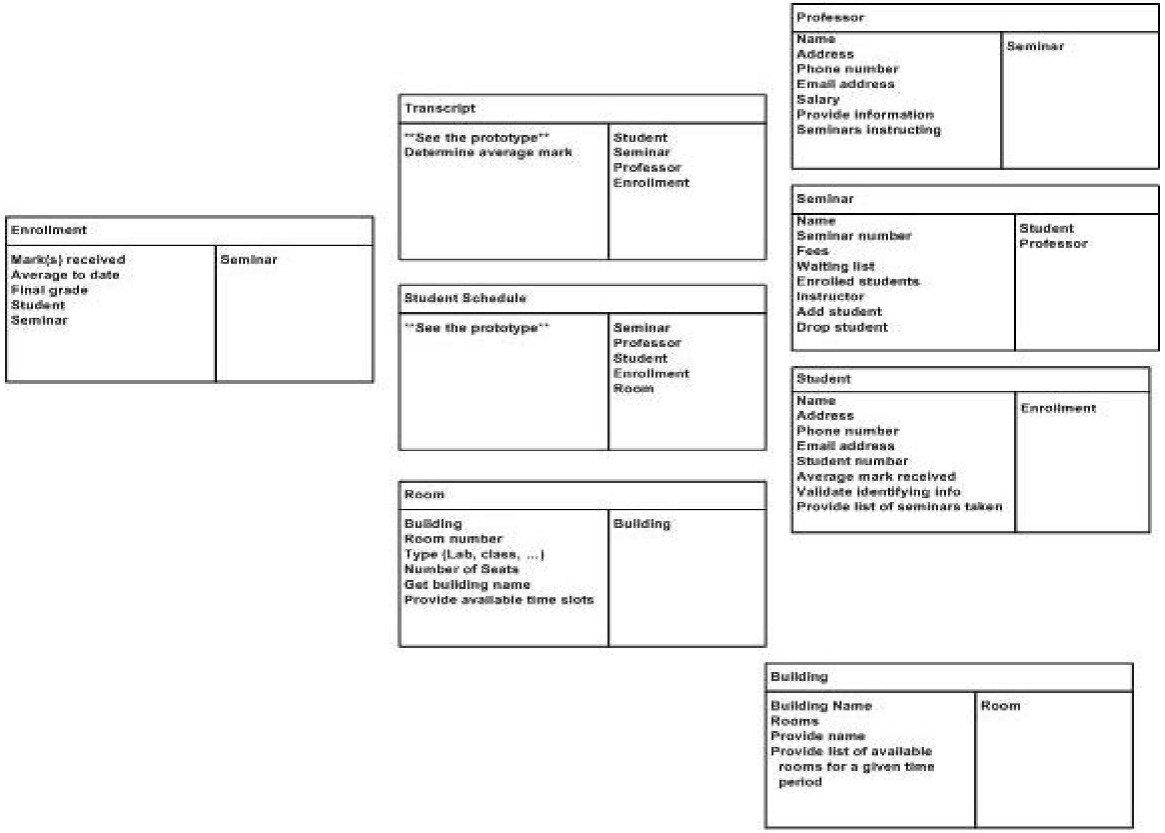


Class Name	
Responsibilities	Collaborators

Example of CRC card



Customer	
Places orders Knows name knows address Knows customer number Knows order history	Order



Enrollment	
Mark(s) received Average to date Final grade Student Seminar	Seminar

Transcript	
See the prototype Determine average mark	Student Seminar Professor Enrollment

Student Schedule	
See the prototype	Seminar Professor Student Enrollment Room

Room	
Building Room number Type (Lab, class, ...) Number of Seats Get building name Provide available time slots	Building

Professor	
Name Address Phone number Email address Salary Provide information Seminars instructing	Seminar

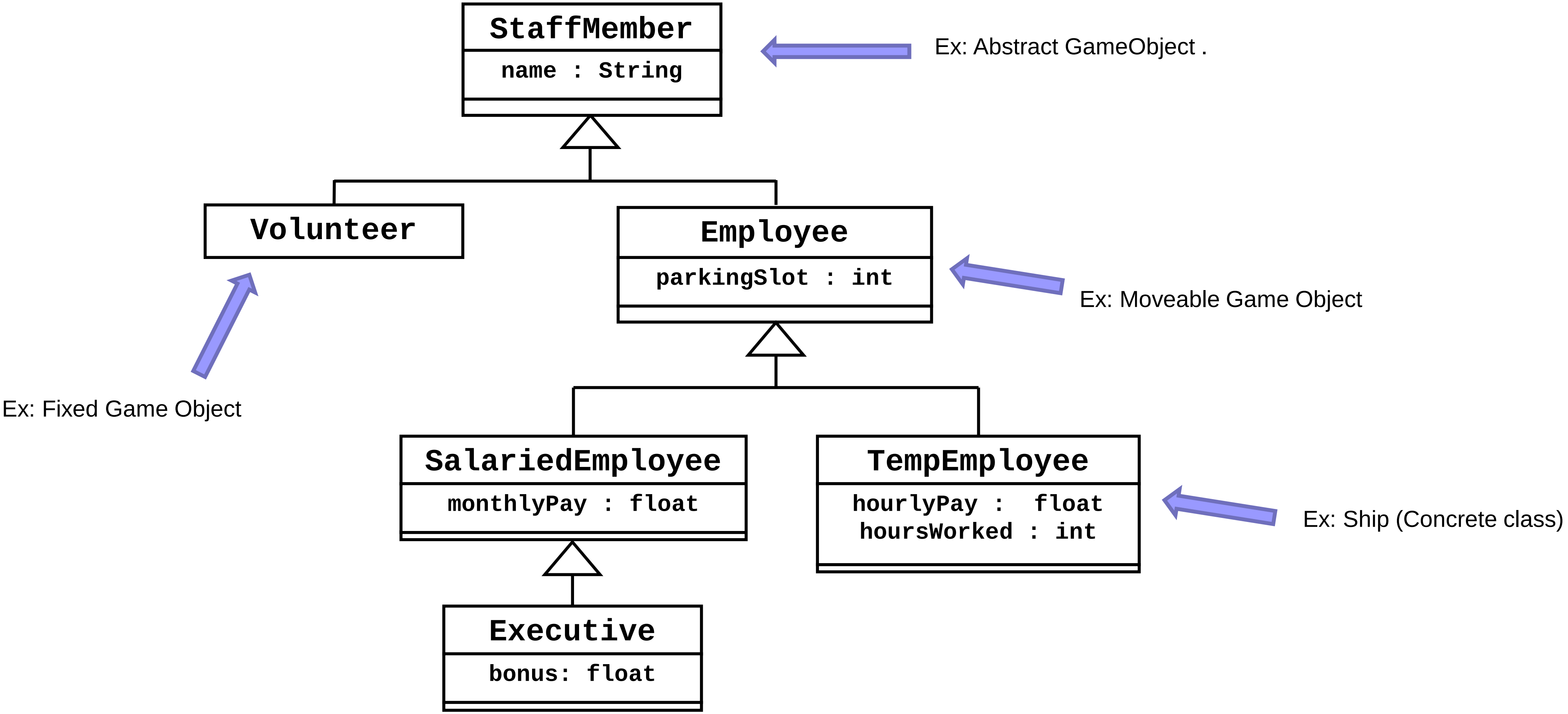
Seminar	
Name Seminar number Fees Waiting list Enrolled students Instructor Add student Drop student	Student Professor

Student	
Name Address Phone number Email address Student number Average mark received Validate identifying info Provide list of seminars taken	Enrollment

Building	
Building Name Rooms Provide name Provide list of available rooms for a given time period	Room

Source: http://agilemodeling.com/artifacts/crcModel.htm

Think sub-classes



StaffMember
name : String

Ex: Abstract GameObject .

Volunteer

Ex: Fixed Game Object

Employee
parkingSlot : int

Ex: Moveable Game Object

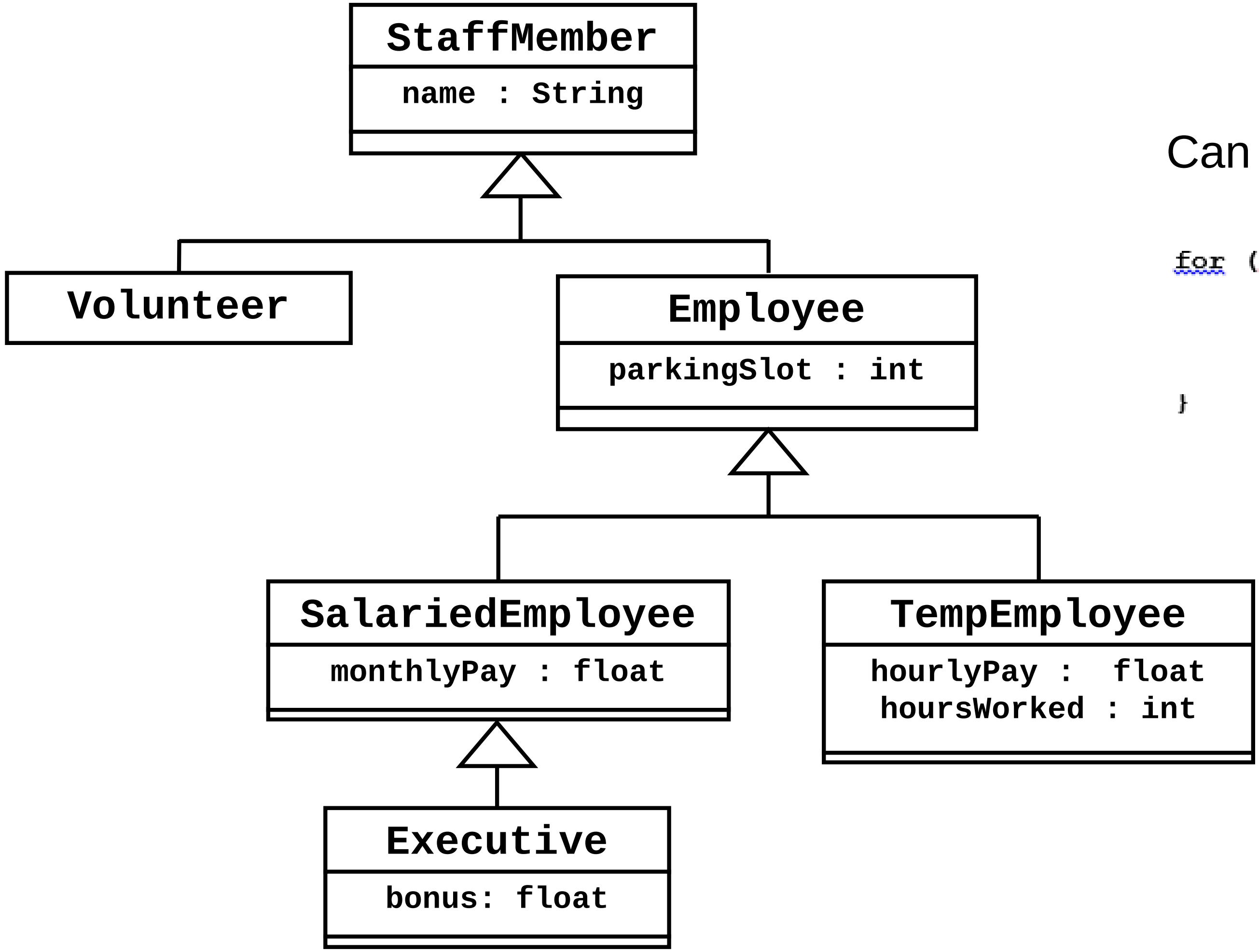
SalariedEmployee
monthlyPay : float

TempEmployee
hourlyPay : float
hoursWorked : int

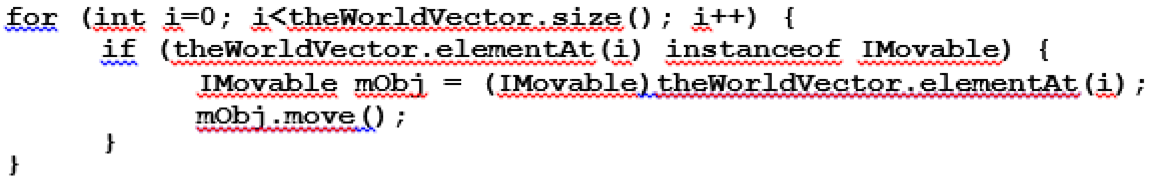
Ex: Ship (Concrete class)

Executive
bonus: float

Think sub-classes (Cont)



Can you work support Runtime Polymorphism safely?



```
for (int i=0; i<theWorldVector.size(); i++) {
        if (theWorldVector.elementAt(i) instanceof IMovable) {
                IMovable mObj = (IMovable)theWorldVector.elementAt(i);
                mObj.move();
        }
}
```

Note: See today lecture on Runtime Polymorphism
And   “Additional details” note in Assignment 1.

Draw your UML Diagram

- Draw your UML in Violet or Creatly
- Check it against the CRC cards and assignment 1 requirements
- Manually execute a few main scenarios
  - Input command ‘a’
  - Input command ‘m’
  - Which classes will do what ?

Coding after completion of UML Diagram

- Code after the UML diagram
  - A decent version
- Expect to return to UML diagram to make changes once identified issues through coding or testing
- Including comments and correct use of variables, class names, package names
- Proper use of inheritance, encapsulation, polymorphism, interface

Coding before UML diagram

- Can I code the entire program without UML diagram?
- Can I generate UML diagram from my Code (code/fix) ?
- Prototyping some key concepts ?
  - I.e. Input processing ?
  - Calling the game object?

Testing the Program

- Validate the complete set of commands
- Check for input errors and boundary conditions
  - Your program should not be crashed under these conditions (software quality)
  - Conform to expected output format (i.e. decimal points)
- Final look at code and comments

Turning the assignment (Suggestions)

- Check the deliverable section of the assignment
- Refresh the dist folder
- Run the command to ensure the program can launch correctly
- Turn in your work before **February 22 before 5 PM**
- Have fun ☺ but not procrastinated