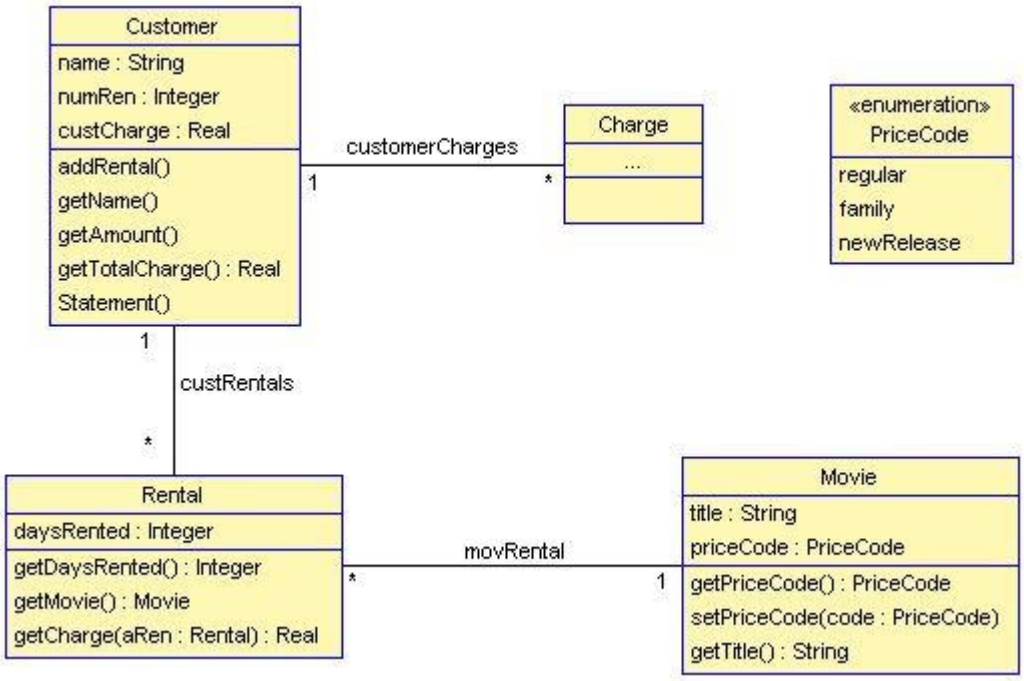
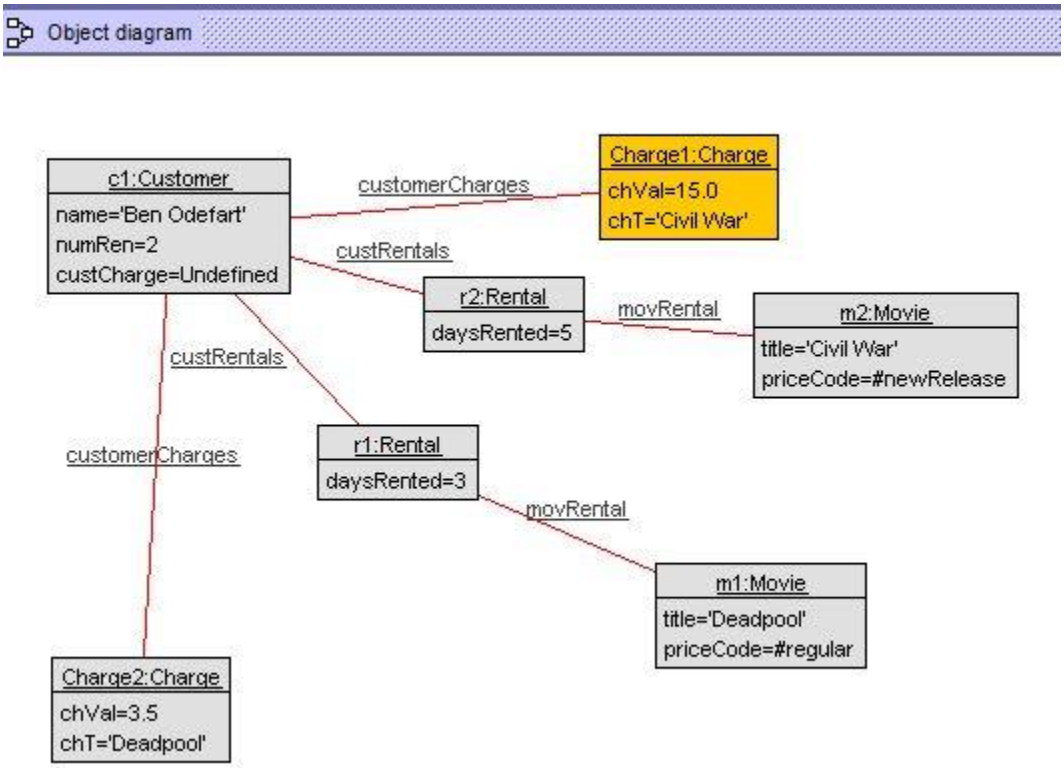
Dan Poss
Tim Wegher
Esof 422

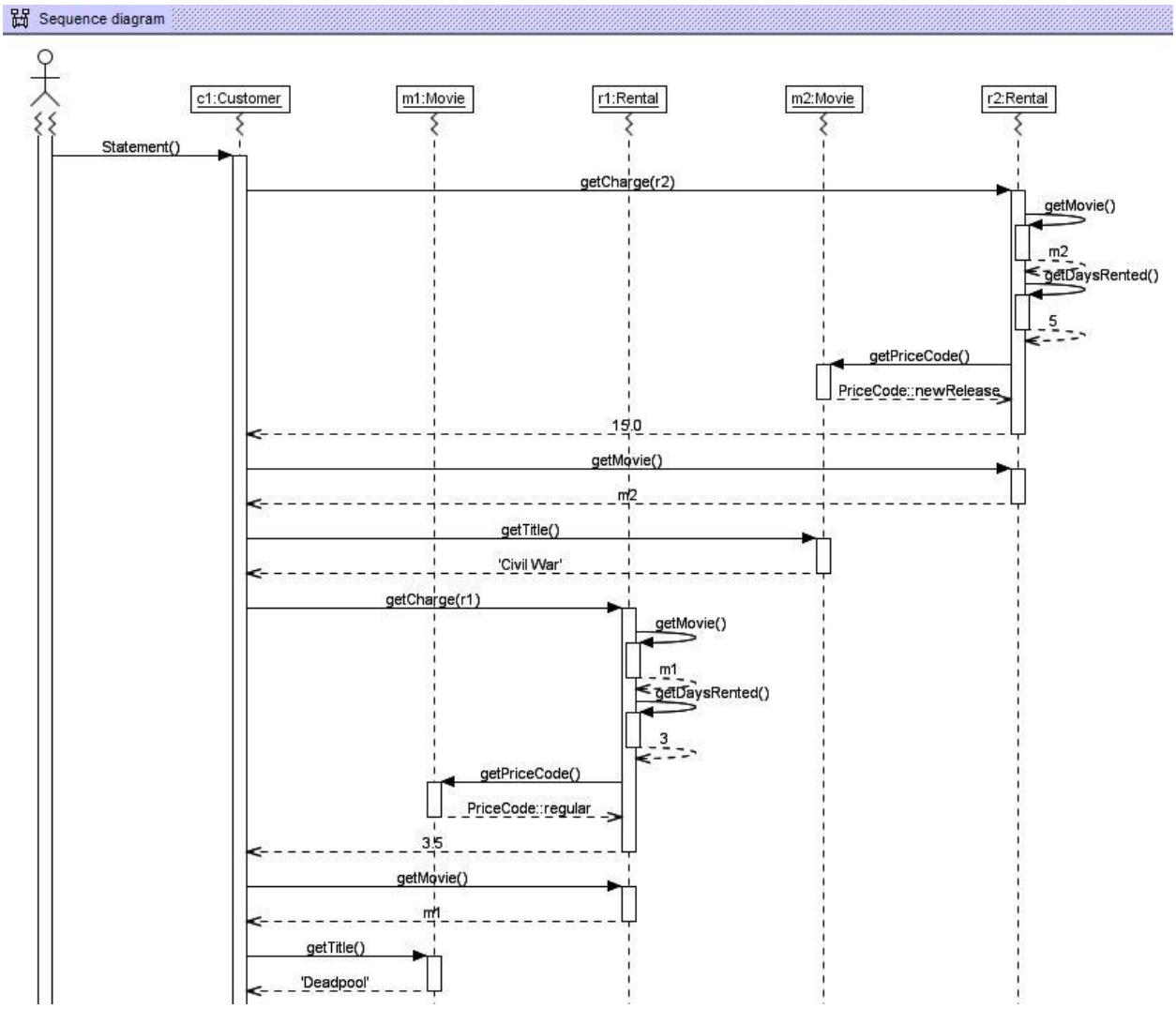
Homework 2

Q1 Movie Rental Model

Class






movierental.use (modified)

--This is a USE model that has embedded SOIL operations in it
--

model MovieRental

enum PriceCode {regular, family, newRelease}

--classes

class Customer
attributes
  name:String
  numRen:Integer
  custCharge:Real

operations
  addRental()
    begin

```
    end
  getName()
  getAmount()
    begin
    end
  getTotalCharge():Real
    begin
      declare ren:Rental;

      self.custCharge:=ren.getCharge(ren);
      result:=self.custCharge;
    end
  Statement()
    begin
      declare aCharge:Charge, sm:Movie, ch:Real, t:String;

      self.numRen:=self.rentals->size();
      for ren in self.rentals do
        ch:=ren.getCharge(ren);
        sm:=ren.getMovie();
        t:=sm.getTitle();
        aCharge:= new Charge;
        aCharge.chVal:=ch;
        aCharge.chT:=t;
        insert(self,aCharge)into customerCharges
    end
  end
end

class Rental
attributes
  daysRented:Integer

operations
  getDaysRented():Integer
    begin
      result := self.daysRented;
    end
  getMovie(): Movie
    begin
      result := self.movie;
    end
  getCharge(aRen:Rental):Real
    begin
      declare wrkCh:Real, m:Movie, pc:PriceCode,dy:Integer;

      m:=aRen.getMovie();
      dy:=aRen.getDaysRented();
      pc:=m.getPriceCode();
```

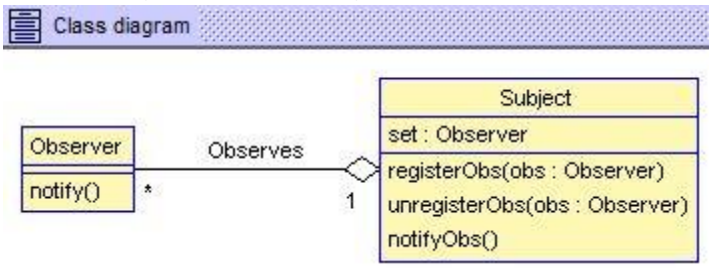
```
    wrkCh:=0;

    if pc=PriceCode::regular then
      wrkCh:=2.0;
      if dy > 2 then
        wrkCh:=wrkCh + (dy -2) * 1.5;
        end;
    end;

    if pc=PriceCode::family then
      wrkCh:=1.5;
      if dy > 3 then
        wrkCh:=wrkCh + (dy -3) * 1.5;
      end;
    end;

    if pc=PriceCode::newRelease then
      wrkCh:=dy * 3.0;
    end;

    result:=wrkCh;
  end
end

class Movie
attributes
  title:String
  priceCode:PriceCode

operations
  getPriceCode():PriceCode
    begin
      result := self.priceCode;
    end

  setPriceCode(code:PriceCode)
    begin
      self.priceCode := code;
    end

  getTitle():String
    begin
      result := self.title;
    end
  end

class Charge
attributes
```

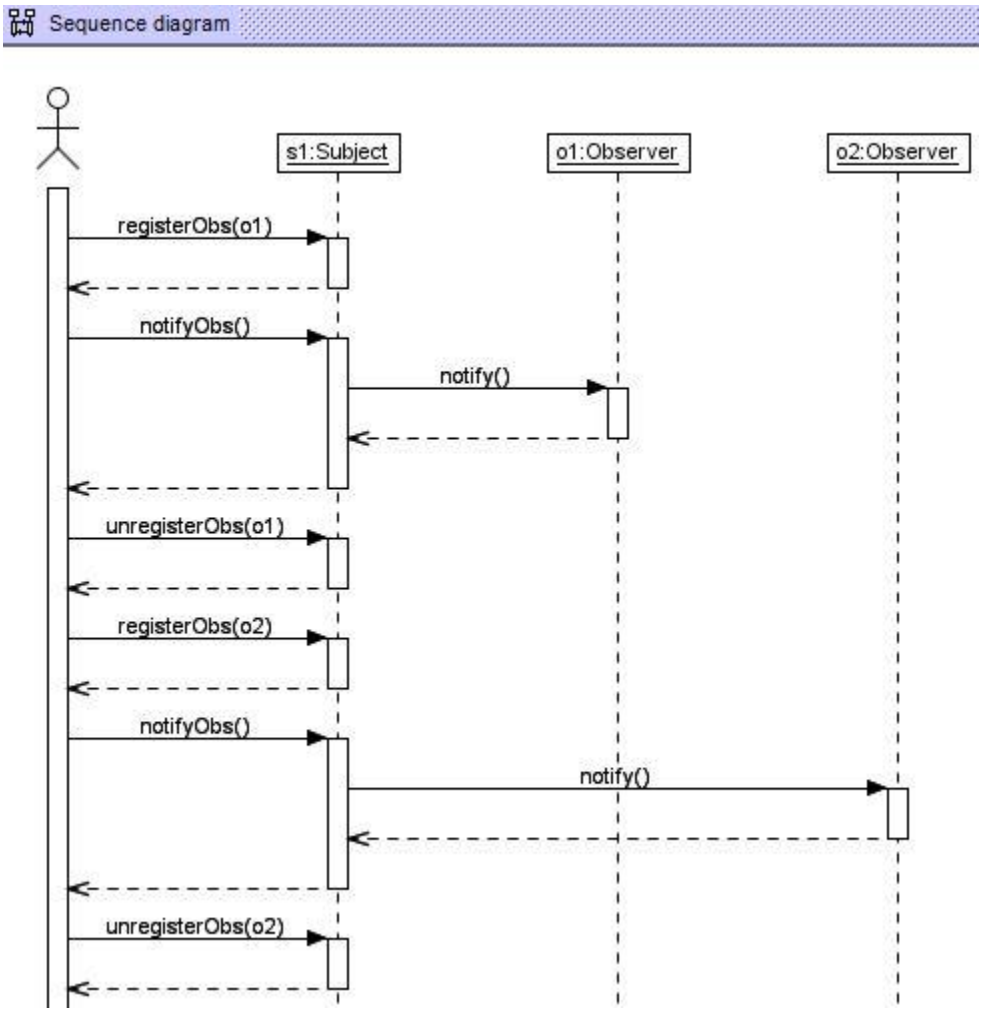
```
  chVal:Real
  chT: String

operations
end

--associations

association custRentals between
  Customer [1] role renter
  Rental [0..*] role rentals
end

association movRental between
  Rental [0..*] role movRentals
  Movie [1] role movie
end

association customerCharges between
  Customer [1] role cust
  Charge [0..*] role charges
end

--constraints
--Added for class exercises

constraints
--Example constraints
--You may remove these constraints in your design.  They are here
--just as examples.

context Customer
  inv maxRental:numRen <= 10
  inv agreement:rentals->size = numRen
  inv rentals:rentals->notEmpty
  inv daysRented:rentals->select(daysRented > 3)->notEmpty
```

movierental.txt (commands used)

```
use> check
checking structure...
checked structure in 2ms.
checking invariants...
checking invariant (1) `Customer::agreement': OK.
checking invariant (2) `Customer::daysRented': OK.
checking invariant (3) `Customer::maxRental': OK.
checking invariant (4) `Customer::rentals': OK.
checked 4 invariants in 0.006s, 0 failures.
use> !create c1:Customer
use> !create m1:Movie
use> !create r1:Rental
use> !insert (c1,r1) into custRentals
use> !insert(r1,m1) into moveRental
<input>:1:0: Association `moveRental' does not exist.
use> !insert(r1,m1) into movRental
use> !create m2:Movie
use> !create r2:Rental
use> !insert (c1, r2) into custRentals
use> !insert (r2, m2) into movRental
use> !set m1.priceCode := PriceCode::regular
use> !setm1.title := 'Deadpool;
<input>:line 1:25 mismatched character '<EOF>' expecting '''
<input>:line 1:0 no viable alternative at input 'setm1'
use> !setm1.title := 'Deadpool'
<input>:1:0: Variable `setm1' in expression `setm1' is undefined.
use> !set m1.title := 'Deadpool'
use> !set r1.daysRented := 3
use> !set m2.priceCode := 2
<input>:1:0: Type mismatch in assignment expression. Expected type `PriceCode', found `Integer'.
use> !set m2.priceCode := PriceCode::newRelease
use> !set r2.daysRented := 5
use> !set m2.title := 'Civil War'
use> !set c1.name := 'Ben Odefart'
use> !set c1.numRen := 2
use> check
checking structure...
checked structure in 1ms.
checking invariants...
checking invariant (1) `Customer::agreement': OK.
checking invariant (2) `Customer::daysRented': OK.
checking invariant (3) `Customer::maxRental': OK.
checking invariant (4) `Customer::rentals': OK.
checked 4 invariants in 0.004s, 0 failures.
use> !c1.Statement()
use>
```

Question 2 (Observer Pattern)




Observer.use

model Observer

class Observer
attributes
operations
 notify()
  begin
   declare awake:Boolean;
   awake:=true;
  end

```
end

class Subject
attributes
  set:Observer
operations
  registerObs(obs:Observer)
    begin
      self.set:=obs;
    end
  unregisterObs(obs:Observer)
    begin
      declare tempObs:Observer;
      self.set:=tempObs;
    end
  notifyObs()
    begin
      self.set.notify();
    end
end

aggregation Observes between
  Subject [1] role Subject
  Observer [0..*] role Observee
end
```

ObserverCMD.x

```
!create s1:Subject
!create o1:Observer
!create o2:Observer
!insert (s1,o1) into Observes
!insert (s1,o2) into Observes

check

!s1.registerObs(o1)
!s1.notifyObs()
!s1.unregisterObs(o1)
!s1.registerObs(o2)
!s1.notifyObs()
!s1.unregisterObs(o2)
```