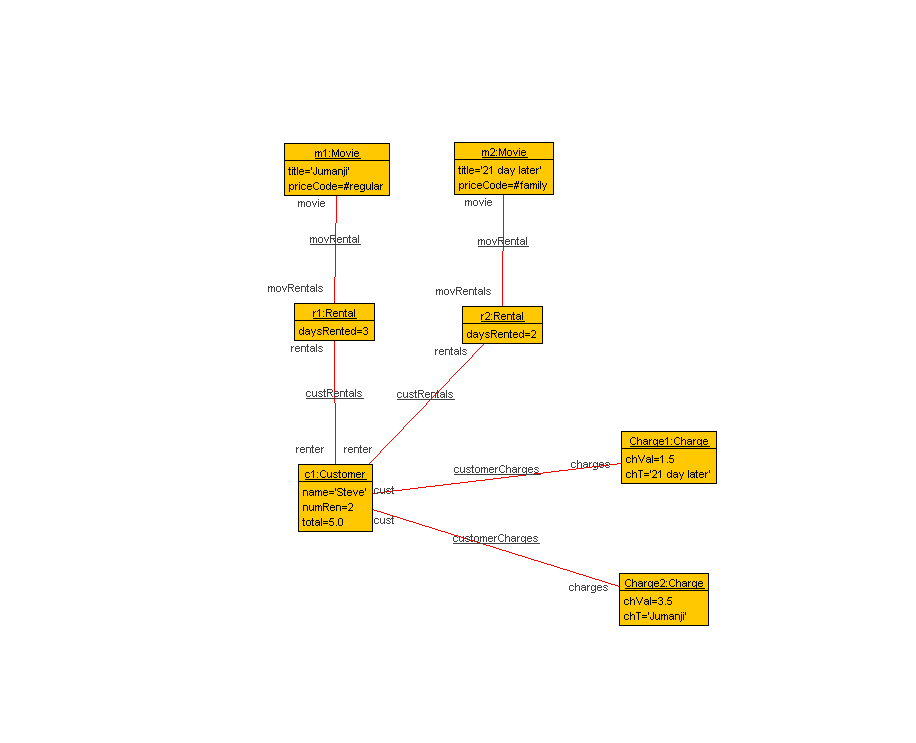
Shelby Huston

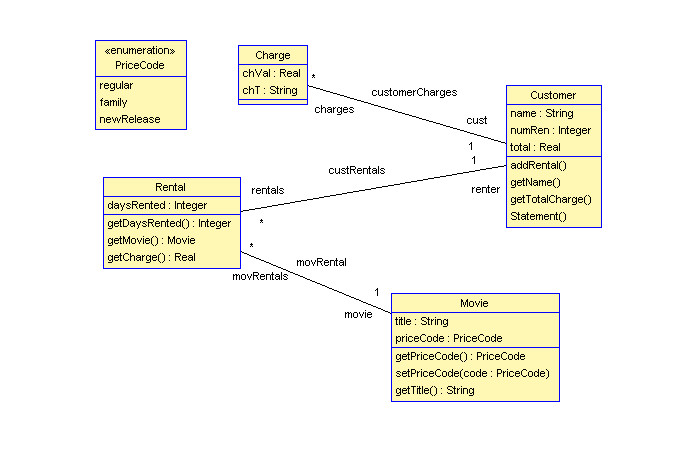
Logan Shy

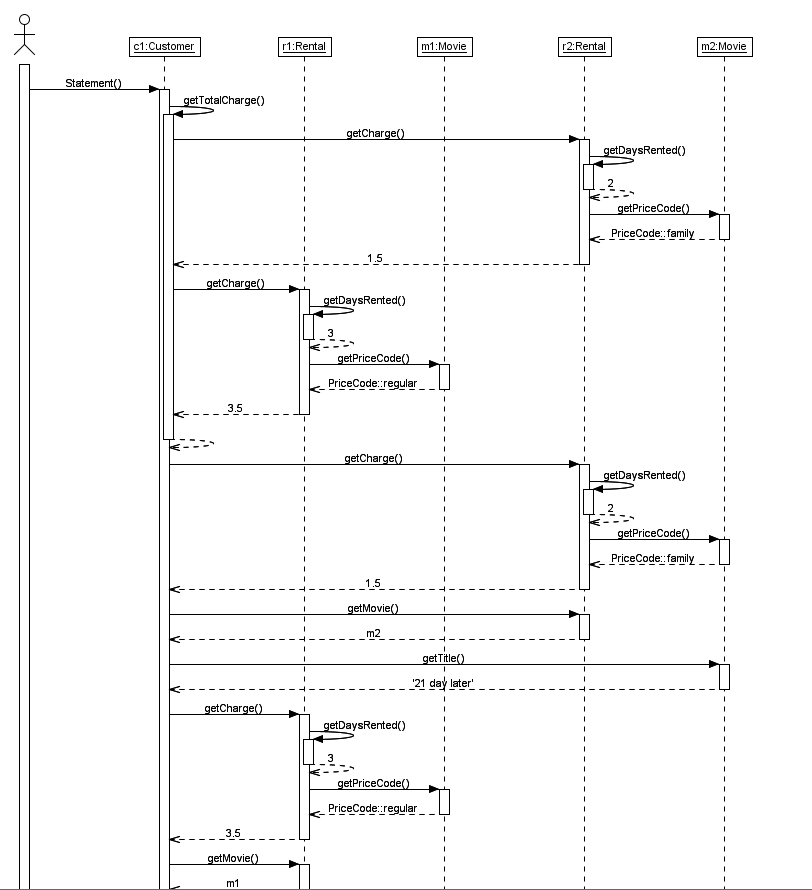
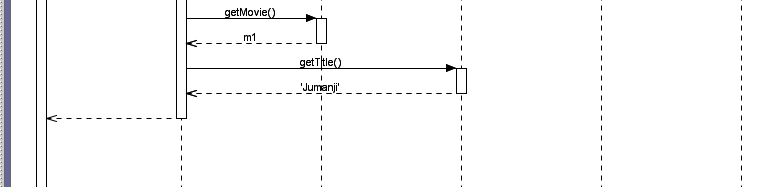
ESOF 422

1/30/20

HW2

1)





--the use file

--Question 1

model MovieRental

enum PriceCode {regular, family, newRelease}

--classes

class Customer

attributes

name: String

numRen: Integer

total: Real

operations

addRental()

begin

end

getName()

getTotalCharge()

begin

declare temp:Real;

self.total:=0;

for ren in self.rentals do

temp := ren.getCharge();

self.total := self.total + temp;

end;

end

Statement()

begin

declare aCharge:Charge, sm:Movie, ch:Real, t:String;

self.getTotalCharge();

self.numRen:=self.rentals->size();

for ren in self.rentals do

ch:=ren.getCharge();

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(): Real

begin

declare wrkCh:Real, m:Movie, pc:PriceCode, dy:Integer;

m:=self.movie;

dy:=self.getDaysRented();

pc:=self.movie.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

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

--.x file

!create c1:Customer

!set c1.name:='Steve'

!set c1.numRen:=2

!create r1:Rental

!set r1.daysRented:=3

!create m1:Movie

!set m1.title:='Jumanji'

!set m1.priceCode:= PriceCode::regular

!create r2:Rental

!set r2.daysRented:=2

!create m2:Movie

!set m2.title:='21 day later'

!set m2.priceCode:= PriceCode::family

!insert (r1,m1) into movRental

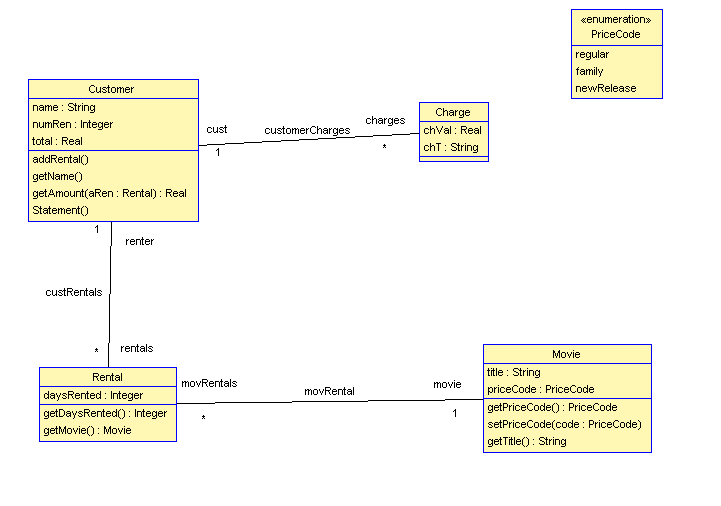
!insert (r2,m2) into movRental

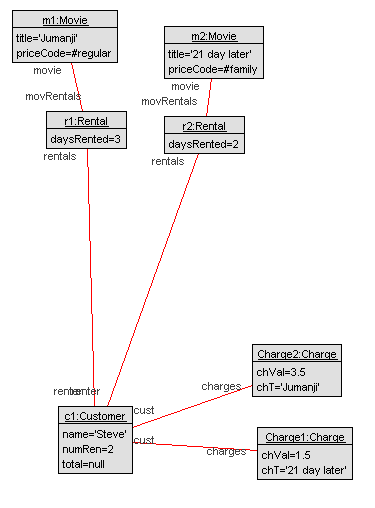
!insert (c1,r1) into custRentals

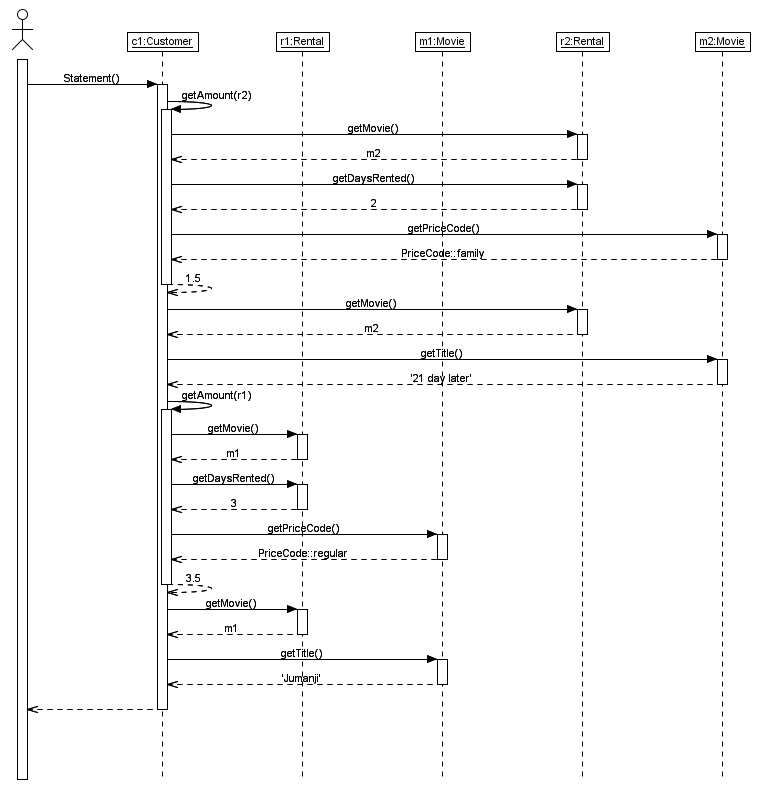
!insert (c1,r2) into custRentals

! c1.Statement()

2)







--Question 2

model MovieRental

enum PriceCode {regular, family, newRelease}

--classes

class Customer

attributes

name: String

numRen: Integer

total: Real

operations

addRental()

begin

end

getName()

getAmount(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

Statement()

begin

declare aCharge:Charge, sm:Movie, ch:Real, t:String;

self.numRen:=self.rentals->size();

for ren in self.rentals do

ch:=self.getAmount(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

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

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

context Customer

inv leastRental:numRen >= 1

inv lateRentas:rentals->select(daysRented > 3) -> notEmpty

--.x file

!create c1:Customer

!set c1.name:='Steve'

!set c1.numRen:=2

!create r1:Rental

!set r1.daysRented:=3

!create m1:Movie

!set m1.title:='Jumanji'

!set m1.priceCode:= PriceCode::regular

!create r2:Rental

!set r2.daysRented:=2

!create m2:Movie

!set m2.title:='21 day later'

!set m2.priceCode:= PriceCode::family

!insert (r1,m1) into movRental

!insert (r2,m2) into movRental

!insert (c1,r1) into custRentals

!insert (c1,r2) into custRentals

! c1.Statement()

--LoyaltyProgram Questions

1) context LoyaltyAccount

init: points:=0

init: membership.card.valid:=true

2) context LoyaltyProgram

inv partners.deliveredServices->size()>1

3) context LoaylatyProgeram::getServices():Set(Service)

body: partners.deliveredServices->asSet()

4) context LoyaltyProgram::getServices(pP:ProgramPartner):Set(Services)

body: if partners-> includes(pP) then

pp.deliverdServices

else

Set{}

endif

5) context Service

inv pointsEarned>pointsBurned

context ProgramPartner

inv deliveredServices->forAll(s:Service|s.pointsEarned>s.pointsBurned)

6) context LoyaltyProgram

Inv partners.deliveredServices->size()=0 -> implies Membership.account.isEmpty=True

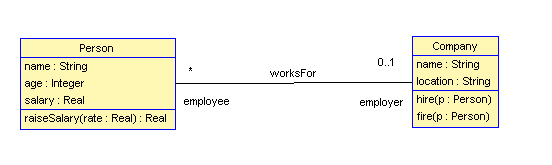
7) contextCustomer::Birthday()

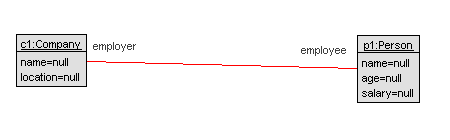
pre:--

post:age:= age@pre + 1

--end LoyaltyProgram Questions

3)





!create c1:Company

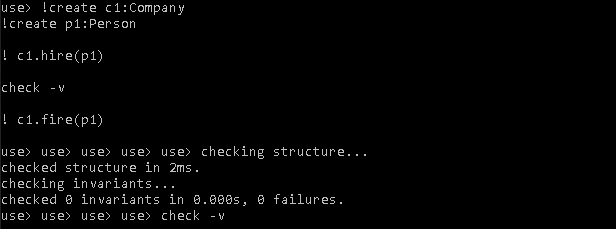
!create p1:Person

! c1.hire(p1)

check -v

! c1.fire(p1)

check -v



Pre and Post conditions pass when they should

!create c1:Company

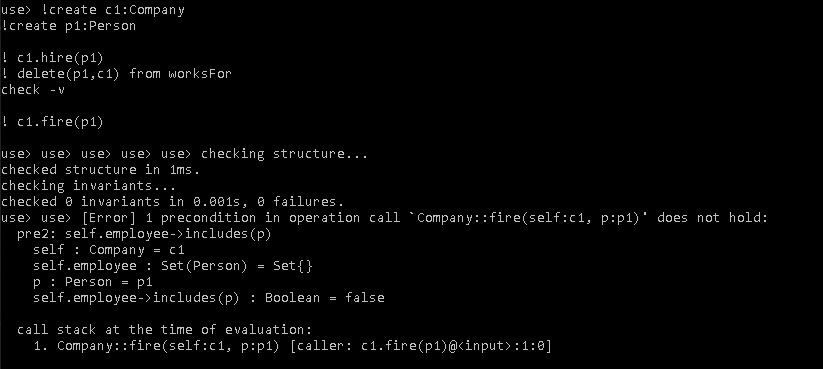
!create p1:Person

! c1.hire(p1)

! delete(p1,c1) from worksFor

check -v

! c1.fire(p1)

check -v

Breaks when we break on purpose