Design a course management system (Like Canvas)

-Professor
  Data: specificCourse, name, loginCredentials
  Behaviors: uploadAssignments, logIn, scoring, writeReview, teaching, uploadRecording
-Students
  Data: name
  Behaviors: submitAssignments, takeCourse, watchRecording
-Course
  Data: name, type, credits, assignments, grade
  Behaviors:
-Internet
  Data: data
  Behaviors: transferData

Sequence of invoking behaviors on objects

```
Professor Siva;
Student Amy;
Internet WaveG;

Siva.logIn(loginCredentials);
Course 5100 = Siva.teaching(name, type, credits, assignments);
Amy.takeCourse(5100);
Siva.uploadRecording(5100);
Amy.watchRecording(5100);
Siva.uploadAssignments(5100);
if Amy.submitAssignments(5100);
   Siva.scoring(5100.assignments);
   if Siva is satisfied with Amy’s assignment
      Siva.writeReview(“good job”);
      else
         Siva.writeReview(“You can do better”);
else
   Siva.scoring(0);
  WaveG.transferData(5100.grade, Amy);
```

Design a pet adoption platform

-Adopter
   Data: name, address, phone
   Behaviors: adoptPet, search, checkOut, writeReview, requestReturnOrder
-Pet
   Data: type, breed, gender, age, furColor
   Behaviors: meow, eat
-Online platform
   Data: computer
   Behaviors: receiveOrder, sendToShipper, returnedByAdopter
- Courier:
   Data: Name,
   Behaviors: deliverPet, contactAdopter

Sequence of invoking behaviors on objects

```
Adopter Lisa;
Courier Wang;
onlinePlatform Adoptapet;
Pet cat1 = Lisa.search(cat, Persian, male, 1, white);
Pet dog1 = Lisa.search(dog, Afador, male, 2, black);
if Lisa find a satisfied pet
   Lisa.adoptPet(cat1);
   Lisa.adoptPet(dog1);
   Lisa.checkOut(Lisa.address, Lisa.phone);
   Adoptapet.receiveOrder(Lisa, cat1);
   Adoptapet.receiveOrder(Lisa, dog1);
   Adoptapet.sendToShipper(Lisa, cat1 and dog1);
   Wang.deliverPet(cat1 and dog1, Lisa.address);
   Wang.contactAdopter(Lisa);
   if Lisa is satisfied with the cat1
      Lisa.writeReview(“So cute”);
   If Lisa is unsatisfied with the dog1
      Lisa.writeReview(“This dog eats tooo much!”);
      Lisa.requestReturnOrder(dog1);
      Adoptapet.returnedByAdopter(Lisa, dog1);
else
   Lisa does not find a satisfied pet on the Adoptapet website.
```

Design an app to book airline ticket.

-Traveler
  Data: name, gender, age, phone, ID, credit card, loginCredentials, emailAddress
  Behaviors: signIn, buy, search
-OnlineApp
  Data: computers
  Behaviors: checkOut, receiveOrder, sendToAirlineCompany, refund, makeETicket, sendReceipt
-Flight
  Data: time, startLocation, destination
  Behaviors:
-Airline Company
  Data: name, flight
  Behaviors: updateAvailableSeats, updateCancellation

Sequence of invoking behaviors on objects

```
Traveler Mark;
OnlineApp Expedia;
AirlineCompany AA;
Mark.logIn(loginCredentials);
Flight aToB1 = Mark.search(9.24, Seattle, San Jose);
if Mark find a satisfied flight
  Mark.buy(aToB1);
  Expedia.checkout(Mark.ID, Mark.phone, Mark.credit card, Mark.emailAddress);
  Expedia.receiveOrder(Mark);
  Expedia.sendReceipt(Mark.emailAddress);
  Expedia.makeETicket(Mark);
  Expedia.sendToAirlineCompany(Mark, AA);
  AA.updateAvailableSeats(aToB1);
  if the flight is canceled
    AA.updateCancellation(aToB1, Expedia);
    Expedia.refund(aToB1, Mark);
    Flight aToB2 = Mark.search(9.25, Seattle, San Jose);
    Mark.buy(aToB2);
    Expedia.checkout(Mark.ID, Mark.phone, Mark.credit card, Mark.emailAddress);
    Expedia.receiveOrder(Mark);
    Expedia.sendReceipt(Mark.emailAddress);
    Expedia.makeETicket(Mark);
    Expedia.sendToAirlineCompany(Mark, AA);
    AA.updateAvailableSeats(aToB2);
else
  Mark does not find the desired flight.
```

Design a course registration platform.

-Student:
   Data: name, ID, emailAddress, loginCredentials
   Behaviors: registerCourse, search, joinWaitlist, cancelRegistration, logIn
-Professor:
   Data: name
   Behaviors: uploadCourse
-Course:
   Data: name, type, subject, credit
-Registration platform
   Data: computers
   Behaviors: setSeats, setRestrictions, sendNotification, updateSeats, pendingRegistration, registrationcanceled

Sequence of invoking behaviors on objects

```
Student Yee;
Professor Siva;
Course INFO5100;
RegistrationPlatform Banner;
Siva.uploadCourse(INFO5100);
Banner.setSeats(INFO5100);
Banner.setRestrictions(INFO5100);
Yee.login(loginCredentials);
Yee.search(Siva, compulsory, CSE, 4);
if seats are available and restrictions are met
   Yee.registerCourse(INFO5100);
   Banner.sendNotification(Yee.emailAddress);
   Banner.updateSeats(INFO5100);
if seats are unavailable
   Yee.joinWaitlist(INFO5100);
   Banner.pendingRegistration(INFO5100);
   if seats are available again
   Banner.sendNotification(Yee.emailAddress);
   Yee.registerCourse(INFO5100);
if restrictions are not met
   Banner.registrationcanceled(Yee, INFO5100);
if Yee is not satisfied with this course
   Yee.cancelRegistration(INFO5100);
   Banner.updateSeats(INFO5100);
```

Order food in a food delivery app.(Like Uber Eats)

-Customer:
 Data: name, emailAddress, loginCredentials, address, phone, credit card
 Behaviors: logIn, search, buy, writeReview, requestCancelOrder, requestRefund, writeReview
-Food delivery app:
 Data: Couriers, computers
 Behaviors: allocateCourier, Ship, sendReceipt, checkOut, refund, sendToMerchant
-Courier:
 Data: Name,
 Behaviors: deliverFood, contactCustomer
-Merchant:
 Data: openTime, type
 Behaviors: makeFood

Sequence of invoking behaviors on objects

```
Customer: Tina;
FoodDeliveryApp: UberEats;
Tina.logIn(loginCredentials);
Merchant KFC = Tina.search(fastfood);
Tina.buy(KFC);
UberEats.checkOut(Tina.address, Tina.phone, Tina.creditCard);
UberEats.sendReceipt(Tina.emailAddress);
if Tina change her mind
    Tina.requestCancelOrder(KFC);
    UberEats.refund(KFC, Tina);
else
  UberEats.sendToMerchant(KFC);
  KFC.makeFood(Tina);
  UberEats.ship(KFC);
  Courier Jina = UberEats.allocateCourier;
  Jina.deliverFood(KFC, Tina.address);
  Jina.contactCustomer(Tina);
  if Tina is satisfied with the food
     Tina.writeReview("So delicious");
  else
      Tina.writeReview("ooooo");
      Tina.requestRefund(KFC, UberEats);
      UberEats.refund(Tina);
else KFC is closed.
```