Problem: Order a hotel online before a trip

Identify Objects and Behaviors:

Thing (Object)/Entity:
Information (Data)/(State)
Services (Behaviors)/Actions

Consumer (Online Shopper):
Data: name, date of birth, phone, email address
Behaviors: reserve, search, reviews, compare, cancel

Internet:
Data: Booking, Airbnb, Expedia, Hotels, Kayak
Behaviors: searchForHotelWebsites

HotelWebsite:
Data: URL, Hotels[] hotels, BankAccount
Behaviors: search, sort, display, compare, place

Hotel:
Data: hotel name, location, price, check-in date, check-out date, number of people will check in, size, rating
Behaviors:

CreditCard:
Data: number, name, company, expire date, security code
Behaviors:

CreditCardCompany:
Behavior: authorizeTransaction

Sequence of invoking behaviors on Objects:

OrderHotelOnline:

Consumer allen,
Internet internet,
HotelWebsite booking,
Hotel myHotel,
CreditCard card,
CreditCardCompany discover,
ReservingConfirmation response,

If Internet.isAvailable
Allen.searchInInternet -> internet, question: Collection of HotelWebsite

```
            pageNumber = 1;
            Loop
                if allen.findsNoPages
                    break
                end
                allen.findDesirableWebsiteInAPage -> internet, question, page Number:
website
                booking = website;
                if booking is not empty
                    break
                else
                    pageNumber += 1
                end
            End
            booking =website
            if booking is not empty or booking != null
                booking.searchForHotel -> priceRange, size, availableDate, location:
Collection of Hotel
                myHotel = hotel
                allen.reserveHotel -> myHotel, creditCard, address, booking:
ReservingConfirmation
                response = reservingConfirmation
            Else
                allen.cantReserveHotel
        Else
            allen.browseInternetAfterAWhileBack
```

Problem: Design an app for calling taxis (e.g. Uber)

Identify Objects and Behaviors:

Thing (Object)/Entity:
Information (Data)/(State)
Services (Behaviors)/Actions

AppForRideService:
Data: name, phone, email address, location
Behaviors: connect, authorize, sendLocation

Consumer:
Data: number, name, bankaccount
Behaviors: book, select plan, reviews, cancel, logInToApp, logOutApp, pay, disconnect

Driver:
Data: number, name
Behaviors: receive, reviews, cancel, logInToApp, driveToDestination

CreditCard:
Data: number, name, company, expire date, security code
Behaviors:

CreditCardCompany:
Behavior: authorizeTransaction

Sequence of invoking behaviors on Objects:

Consumer allen
Driver adam
AppForRideService uber
CreditCard card,
CreditCardCompany discover,
ReservingReciept receipt,

allen.loginToAppForRideService -> uber: authorize
if (authorize is true)
allen.selectPlan -> uber
allen.connect -> uber, adam: connected
if connected is true
Loop
allen.sendLocation -> uber, adam, location: response
adam.receive -> uber, allen, location: response

```
                    if allen.wantsToCancel or adam.wantsToCancel
                        allen.cancel -> uber
                        adam.cancel -> uber
                    end

                    if (adam.driveToDestination = true)
                        allen.pay -> creditCard, uber: receipt
                    end

            End
else
    allen.cantConnectWithUber
end
```

Problem: Design a job searching and posting platform

Identify Objects and Behaviors:

Thing (Object)/Entity:
Information (Data)/(State)
Services (Behaviors)/Actions

JobSeeker:
Data: name, location, email, phone number, resume
Behavior: search, uploadOrEditInfo, uploadResume

Employer:
Data: job title, job description, salary, working hours, requirement
Behavior: post, receive

Job:
Data: title, date, address, salary, requirement, description

Platform:
Data: JobSeeker data and job data
Behavior: match, authorize

Sequence of invoking behaviors on Objects:

Job searching:

```
JobSeeker allen;
Employer google;
Platform 58tc;
Job myJob;
JobApplying response;

allen.logInThePlatform -> 58tc: authorize
if (authorize is true)
    allen.uploadOrEditInfo -> put name, location, email, phone number, resume into
database
    allen.searchJob -> 58tc: Collections of jobs
    PageNumber = 1;
    Loop
        if allen.findsNoPages
            break
        end

        allen.findDesirableJob -> 58tc, page Number: job
        myJob = job;
```

```
            if myJob is not empty
                break
            else
                pageNumber += 1
            end
        End
        myJob = job
        allen.uploadResume -> myJob, resume: JobApplying
        response = JobApplying
```

Job posting:

```
    Employer google;
    Platform 58tc;
    Job wantedJob;
    JobPosting response;


    google.logInThePlatform -> 58tc: authorize
    if (authorize is true)
        Loop
            google.post -> job title, job description, salary, working hours,
            requirement: job
        End
        wantedJob = job
        if 58tc.match -> resume, wantedJob: response
        else
            58tc.cantMatch
    Else
        google.needReAuthorize
```

Problem: Order food in a restaurant

Identify Objects and Behaviors:

Thing (Object)/Entity:
Information (Data)/(State)
Services (Behaviors)/Actions

Consumer:
Data: name
Behavior: pay, eat, authorize, selectFood, checkMenu, goToRestaurant, findAnotherRestaurant

Restaurant:
Data: address, menu, opening hours
Behavior: getOrder, makeFood, open, close

Menu:
Data: name, price, picture

Food:
Data: dishes

CreditCard:
Data: number, name, company, expire date, security code
Behaviors:

CreditCardCompany:
Behavior: authorizeTransaction

Sequence of invoking behaviors on Objects:

```
consumer allen;
restaurant mcdonalds;
Menu menu;
Food dishes;
CreditCard card,
CreditCardCompany discover,
ReservingReciept receipt

if (mcdonalds.open == true)
        allen.goToRestaurant  -> restaurant
        allen.checkMenu -> menu
        number = 0;
        Loop
                allen.selectFood -> dishes
```

```
            number++
            end
        End
        if number != 0
            mcdonalds.makeFood -> dishes: Collections of food
        end
        allen.eat -> dishes: response
        allen.pay -> creditCard, restaurant: receipt
else
        allen.findAnotherRestaurant
```

Problem: Design a course registration platform

Identify Objects and Behaviors:

Thing (Object)/Entity:
Information (Data)/(State)
Services (Behaviors)/Actions

Student:
Data: name, email address, studentId
Behaviors: register, search, cancel, waitForAvailable

Professor:
Data: name. room, time, location
Behaviors: post, receive, waitForAvailable

University Website:
Data: name, info
Behaviors: isAvailable

Courses:
Data: name, location, time, availability, room

Sequence of invoking behaviors on Objects:

Register classes:

```
Student allen,
UniversityWebsite neu,
Courses myCourse,
RegisterConfirmation response,

If neu.isAvailable
    allen.searchInUniversityWebsite -> neu, name: Collection of Courses
    pageNumber = 1;
    Loop
        if allen.findsNoPages
            break
        end
        allen.findDesirableCoursesInAPage -> neu, name, page Number: course
        myCourse = course;
        if myCourse is not empty
            break
        else
            pageNumber += 1
        end
```

```
            End
            myCourse = course;
            if( availability == true)
                allen.register -> myCourse, studentId: RegisterConfirmation
                response = RegisterConfirmation
            else
                allen.cantRegisterYet
        Else
            allen.waitForAvailable

Post classes:
        Professor wang;
        UniversityWebsite neu,
        Courses myCourse,
        If neu.isAvailable
            wang.logInUniversityWebsite -> neu: authorize
                if (authorize is true)
                    Loop
                        wang.post -> name, location, time, availability, room:
                        course
                    End
                        myCourse = course

                else
                    wang.needReAuthorize
        else
            wang.waitForAvailable
```