INFO 5100 Assignment 1
JINRAN TIAN

1)Order a hotel online before a trip

Object and behaviors:

Consumer(internet hotel booker)
Data: Name, Email, Phone
Behaviors: book, search, reviews, compare, cancel

Internet
Data: HotelsCombined, Trivago, Booking, Hotels, Orbitz,
: Group of websites(Collection of websites)
Behaviors: searchForHotelWebsites

Hotelbookingwebsite
Data: URL, Hotel[] hotels, BankAccount
Behaviors: search, sort, display, compare, bookTheHotel

Hotel
Data: Price, duration, location, hotelratings, brands
Behavior:

CreditCard
Data: Number, name, company, expiry, security code
Behavior:

CreditCardCompany
Behavior: authorizeTransaction

Sequence of Flow - Invoke Objects with Behaviors

Consumer terry;
Internet internet;
ElextronicWebsite orbitz;
Hotel holidayInn;
CreditCard card;
CreditCardCompany visa;
ShoppingConfirmation response;

If Internet.isAvailable
terry.searchInInternet -> internet, question: Collection of HotelbookingWebsite

```
        //terry.findDesirableWebsiteInFirstPage -> Collection of Websites : website
        pageNumber = 1;
        Loop
            If terry.findsNopages
                break
            end
            terry.findDesirableWebsiteInAPage -> internet, question, page Number :
website
            orbitz = website;
            If orbitz is not empty
                break
            else
                pageNumber = pageNumber + 1
            end
        End
        orbitz = website
        If orbitz is not empty or orbitz!=null
            orbitz.searchForHotel -> priceRange, duration, location,brand : Collection
            of Hotel
            holidayInn = hotel
            terry.bookHotel -> holidayInn, creditCard, email, orbitz:
BookingConfirmation
            response = bookingConfirmation
        Else
            terry.cantBookHotel
    Else
        terry.browseInternetAfterAWhileBack
```

2)design an app for calling taxis(e.g. uber)

objects and behaviors:

TaxiCallingApp
    Data:name, phoneNumber
    Behavior: receivePassengerSignal, pairDriversToPassenger, sendDriverSignal,authorize

Passenger
    Data:number, name
    Behavior: loginToApp, sendPassengerSignal,uploadLocation, uploadDestiny,
stopSendingSignal

TaxiDriver
    Data:name,number,carType,rating,status

Behavior:loginToApp, receiveRequestSignal, pickUpPassenger, driveToDestiny

Sequence of Flow - Invoke Objects with Behaviors

```
Passenger terry;
Driver kyle;
TaxiCallingApp uber;
terry.loginToTaxiCallingService -> Uber : authorize
if (authorize is true)
        terry.sendPassengerSignal -> Uber, currentLocation,destiny : waitingTimeInfo
        if waitingTime < 10min
                Uber.pairDriversToPassenger-> terry, currentLocation,destiny : kyle
                if kyle.status = free
                        Uber.sendDriverSignal -> kyle : requestconfirmation
                        kyle.pickUpPassenger-> terry, location, destiny : pickUpConfirmation
                        kyle.driveToDestiny ->terry, destiny : tripConfirmation
                else
                        Uber.findAnotherDriver
        terry.stopSendingSignal
else
        terry.cantCallTaxi
end
```

3) design a job searching and posting platform

objects and behaviors:

jobSearcher
Data: name, phone, address, email, resume
behavior: loginInToJobPlatform, searchJob, applyJob, compare

jobPoster
Data: companyName, phone, address, email, positionTitle
behavior: loginInToJobPlatform, receiveApplication, makeDecision, refuseApplication, acceptApplication

jobPlatform
Data: name, phone, email
Behavior: authorize, connect, postCompanies, postPositions, acceptUploads

Sequence of Flow - Invoke Objects with Behaviors

jobSearcher terry

```
jobPoster Amazon
JobPlatform linkedin
terry.loginToJobPlatform -> linkedin : authorize
if (authorize is true)
        terry.searchJob -> linkedin, softwareEng : Amazon
        terry.applyJob -> resume, Amazon
        Amazon.receiveApplication -> terry, resume : appendApplicationList
        if Amazon.positionAvailble
                Amazon.makeDecision -> terry, resume : result
                if result = Amazon.acceptApplication
                        Amazon.offerInterviews
                else
                        Amazon.refuseApplication
        else
                refuseApplication
else
        terry.cantapplyforjobs
```

4) Order food in a restaurant

objects and behaviors:

Customer
- Data: name, status
- Behavior: readMenu, askQuestion, orderFood

Waiter
- Data: name, status
- Behavior: answerQuestion, recordOrder

Sequence of Flow - Invoke Objects with Behaviors

```
Customer terry
Waiter john
terry.readMenu
if terry.status = ready to order
        If john.status = free
                Loop
                        if terry.haveQuestions
                                terry.askQuestions-> john : question
                                john.answerQuestions -> terry, question : answer
                        terry.orderFood-> john : order
                        john.recordOrder-> terry,order : appendOrderlist
```

```
                    if terry.finishedOrder
                        break
            response =  john.Orderlist
        else
            terry.wait
else
        terry.stillNeedTime
```

5)design a course registration platform

objects and behaviors:

Student
- Data: name, studentID, email.
- Behaviors: search, registrate, drop,loginInToPlatform

Courses
- Data: courseName, courseID, courseDescription, major, professor, status
- Behaviors:

CourseRegistrationPlatform
- Data: name, courseName, studentID,courseNumber
- Behavior: searchForCourses, sort, display, authorize, record,

Sequence of Flow - Invoke Objects with Behaviors

```
Student terry
CourseName INFO5100
CourseRegistrationPlatform NEU
If Internet.isAvailable:
        terry.loginToPhoneService -> NEU : authorize
        if (authorize is true)
            terry.searchForCourses -> NEU, question: Collection of Courses
            Loop
                If terry.findsNoCourses
                    break
                end
                terry.findDesirableCourses -> NEU, courseDescription: CourseName
                    INFO5100 = CourseName
                    If INFO5100.status != full
                        terry.registerCourse -> INFO5100, NEU:
        RegisterConfirmation
                    reponse = registerConformation
```

```
        else
                terry.cantRegisterCourse
else
        terry.registerCourseLater
```