5100 Assignment 1

------ Jin Yang

Questions:

1. Order a hotel online before a trip.
2. Design an app for calling taxis (e.g. Uber).
3. Design a job searching and posting platform.
4. Order food in a restaurant.
5. Design a course registration platform.

1. Order a hotel online before a trip.

Objects and Behaviors:

Traveler

Data: name, destination, dateOfTrip, fellowTraveler, phone, identity card

Behaviors: searchHotels, inputRequirements, reviewInformation, compare, orderHotel, cancelOrder

Internet

Data: Group of booking websites

Behaviors: searchForHotelBookingWebsite

HotelBookingWebsite

Data: URL, Hotel[] hotels, bankAccount

Behaviors: Search, sort, display, compare, placeOrder

Hotel

Data: address, price, bankAccount, facilities, yearOfBuilt, trafficCondition, parking

Behaviors: checkAvailability, reserveRoom

DebitCard

Data: number, nameOnCard, company, expiryDate, securityCode

Behavior:

Invoke Objects with Behaviors:

```
OrderHotelBeforeTrip
    Traveler Jin;
    Internet internet;
    HotelBookingWebsite bookingWebsite;
    Hotel hotel;
    DebitCard card;
    BookingConfirmation response;

    Jin.searchHotels -> collection of HotelBookingWebsite
    if ! Jin.findHotelBookingWebsite
        break;
    Jin.inputRequirements;
    Loop
        bookingWebsite.search;
        if bookingWebsite.findsNoPages
            break;
        end if
        bookingWebsite.display;
        Jin.reviewInformation;
        if Jin.findHotel
            hotel = foundHotel;
            hotel.checkAvailability;
            if hotel.availabile
                break;
            end if
        end if
    end Loop
    Jin.orderHotel -> bookingWebsite, hotel, card, address;
    bookingWebsite.placeOrder;
    hotel.reserveRoom;
```

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

Objects and Behaviors:

Passenger

Data: name, phoneNumber, location, destination, card

Behaviors: login, inputCardInfo, orderTaxi, cancelOrder

App

Data: passengerInfo, TaxiDriverInfo, passengerlocation, driverlocation

Behaviors: receivePassengerOrder, sendOrderToDriver, contactTaxiCompany

TaxiCompany

Data: taxiDriverInfo

Behaviors: checkDriverCredit

TaxiDriver

Data: name, phoneNumber, location, card

Behaviors: login, inputCardInfo, getPassengerInformation, takeOrder, pickupPassenger

Card

Data: number, nameOnCard, company, expiryDate, securityCode

Invoke Objects with Behaviors:

```
AppForTaxi
        Passenger psg;
        TaxiDriver drvr;
        App uber;
        TaxiCompany cabcmpy;
        Card card;

        psg.login;
        psg.inputCardInfo;
        drvr.login;
        drvr.inputCardInfo;
```

```
psg.orderTaxi;
if !psg.cancelOrder
    uber.receivePassengerOrder;
    Loop
        if passengerlocationIsNearToDriverlocation
        uber.contactTaxiCompany;
        if TaxiCompany.checkDriverCredit
            break;
        end if
    end loop
    app.sendOrderToDriver;
    drvr.getPassengerInformation;
    if drvr.takeOrder
        drvr.getPassengerInformation;
 end if
drvr.pickupPassenger;
```

3. Design a job searching and posting platform.

Objects and Behaviors:

Employee

Data: name, phone number, email, abilities, idealJob, idealWorkTime

Behaviors: login, jobSearching, submitResume

Company

Data: name, positions, location, numberOfPositions, hiringTime

Behaviors: login, jobPosting, cacelPost, acceptResume

Job

Data: name, request, postingTime

Behaviors:

Platform

Data: name, URL

Behaviors: authorizeIdentity, checkPositionUpToDate, recommendJob, notifyEmployee

Invoke Objects with Behaviors:

```
JobPlatform
    Employee employee;
    Company company;
    Job job;
    Platform pltfm;

    company.jobPosting;
    Loop
        employee.login;
        if pltfm.authorizeIdentity
            break;
    end loop
    employee.jobSearching;
    // suitable positions will be recommended to employee, here we assume the employee will only take one job
    Loop
    if pltfm.checkPositionUpToDate
        pltfm.recommendJob;
        if employee.submitResume
            break;
        end if
    end if
    if company.acceptResume
    pltfm.notifyEmployee;
```

4. Order food in a restaurant.

Objects and Behaviors:

Customer

Data: name, tableNumber, orders, card

Behaviors: checkMenu, wait, order, eat, advise, pay

Waiter/Waitress

Data: name, busy

Behaviors: takeOrder, serveDish, askForAdvise

Cook

Data: name, busy

Behaviors: receiveOrder, finishcooking

Invoke Objects with Behaviors:

```
OrderFood
    Customer customer;
    Waiter waiter;
    Cook cook;
    customer.checkMenu;
    if waiter.busy
        customer.wait;
    Loop
        customer.order;
        waiter.takeOrder;
        if customer.endOfOrder
            break;
        end if;
    end loop
    if cook.busy
        wait;
    loop
        if cook.cooking
            waiter.serveDish;
        end if;
        if cook.end OfCooking
            waiter.notifyCustomer;
            break;
        end if;
    end loop
    waiter.askForAdvise;
    customer.pay;
```

5. Design a course registration platform.

Objects and Behaviors:

student

Data: name, ID, major

Behaviors: login, search, review, register, drop

course

Data: number, professor, time, semester, availableStudent

Behaviors:

platform

Data: courseInfo

Behaviors: authorize, showCourseInfo, checkTimeConflict

Invoke Objects with Behaviors:

```
CourseRegistrationPlatform
    Student student;
    Course course;
    Platform platform;

    student.login;
    if ! platform.authorize
    loop
      student.login;
    end
    student.search;
    platform.showCourseInfo;
    student.review;
    if ! platform.checkTimeConflict
      if course.availableStudent != 0
            student.registar;
```

```
        course.availableStudent -= 1
    end if
end if
```