1.Problem: Program to order a hotel online before a trip

Objects and Behaviors:

Consumer:
Data: Name, Check-in date, Check-out date, Phone number
Behaviors: Search, Reviews, Reserve, Cancel, Compare

Hotel:
Data: Name, Price, Location, Contact Information, Availability
Behavior: Boolean Confirmation

OrderSystems:
Behaviors: orderHotels;

----------Program----------
Consumer Winifred

```
orderHotels(Consumer  Winifred){
Hotel targets[]=Winifred.search(Check-in date, Check-out date,location);
If (targets[].length()==0){
        return false;
        System.out.println("There are no matches");
}
Else{
        Boolean result=false;
        While(targets.length()>0){
              Hotel target=Wnifred.compare(targets[]);
              Boolean decision=Winifred.reviews(Check-in date, Check-out
              date, target);
              If (decision==true){
                    Winifred.reserve(target);
                    Boolean hotelConfrimation=target.confirmation(Winifred);
                    If (hotelConfirmation==true)

                          result= true;
                          Winifred.reserve(target);break;
```

```
                else
                        targets[].delete(targets);
                        System.out.println("Application has been denied by the
                        hotel,Please choose another one");

        }
        else{
        Winifred.cancel();
        targets[].delete(targets);
        }
        }
if(result==false)
        System.out.println("failed, Please run the program again");
return result;

}
}
```

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

Objects and Behaviors:

Consumer:
Data: Name, Location, Destination
Behaviors: findTaxi(), review(), sendRequests()

Taxi:
Data: Location, Reviews ,Taxiinformation
Behaviors: makeDecision(), acceptOrder(), rejectOrder();

CallingApp
Behavior: callingTaxi()

----------Program----------
Consumer Winifred

callingTaxi(){

```
Taxi[] taxis=Winifred.findTaxi(Winifred.Name,Winifred.Location);
Boolean decision=false;
Taxi targetTaxi;
for(int i=0;i<taxis.length();i++){
    decision=Winifred.review(taxis[i].Locations,taxis[i].Reviews,
    taxis[i].Taxiinformation);
    if(decision==true) {
        Winifred.sendRequests(taxi[i]);
        targetTaxi=taxi[i];
        break;
    }
}
If(decision==false)
    System.out.println("Please find again")
else{
    Boolean feedback=targetTaxi.makeDecision(Winifred);
    If(feedback==true){
```

```
            targetTaxi. acceptOrder(Winifred);
            System.out.println("You have called a taxi successfully")
        }
        Else{
        targetTaxi. rejectOrder(Winifred);
        System.out.println("The order was rejected,please find an new taxi");

        }
}
}
```

3.Problem: Design a job searching and posting platform.

Objects and Behaviors:

Jobseeker:
Data: Name, Resume, Seekingposition
Behaviors: Searchingforjob (), uploadInformation(), contactEmployer()

Job:
Data:Name, Requirement,

Employer:
Data: Wage, jobInformation, Requirement
Behaviors: posting()
Platform:
Behaviors: postOntheplatform(),search(), display()

----------Program----------
Jobseeker Winifred
Employer Aaron
Platform Findajob

posting(){

Job[]jobs=Aaron.posting(Aaron.Wage,Aaron.jobInformation,Aaron.Requirement);
Findajob.postOntheplatform(jobs);

}

Searchingforjob(){

Winifred.uploadInformation(Winifred.Name,Winifred.Resume);
Job[] jobs = Findajob.search(Winifred.Seekingposition);
FindaJob.display();

for(int i=0,i<jobs.length,i++){
Winifred.contactEmployer();
}
}

4.Problem: Order food in a restaurant

Objects and Behaviors:

Consumer:
Data: table number, addfood
Behaviors: orderFood(), skim(),addfoods()

Diningtable:
Data: table number, ordercondition

Restaurant:
Data: Menu,
Behaviors: acceptOrders(), cooking();

Ordersystem:
Behavior: showMenu(), processOrder()

----------Program----------
Consumer Winifred
Consumer Taylor
Ordersystem onlineOrdersystem;
Dinningtable Table502
Restaurant FareStart

processOrder(){
onlineOrdersystem.showMenu();
Winifred.skim();
Winifred.orderfood(Table502.tablenumber);
If(Taylor.addfood==true){
Taylor.addfoods(Table502.tablenumber);
}
FareStart.acceptOrders();
FareStart.cooking();
}

5.Problem: Design a course registration platform.

Objects and Behaviors:

Students:
Data: subject, background,Timetable previousTimetable
Behaviors: studentLogin(), review(),searchCourse(),registerCourses()

Course:
Data: CourseID, Name, meetingTime, Credits, Professor, prerequisiteCourse
Behavior: ……

Registrationplatform:
Data: RegistrationCondition, Creditlimit
Behavior: Registration(),checkTime().qualify()

Timetable:
Data:Time

----------Program----------
Student Winifred
Registrationplatform Courseregistration
Course courses[]

```
Registration(){
int credit=0;
Winifred.studentLogin();
courses[]=Winifred.searchCourse(Winifred.subject);
for(int i=0,i<course.length, i++){
        boolean condition=Winifred.review(courses[i]);
        boolean time= Courseregistration.checkTime(course[i].meetingTime,
        Winifred.previousTimetable);
        if(time==true&&ondition==true&&
        Courseregistration.qualify(course[i]. prerequisiteCourse,
        Winifred.background) ==true){

            if(credit+course[i].Credits > Courseregistration.creditLimit)
                system.out.println("you have met the credits limitation")
            else
                Winifred.registerCourse(course[i]);
```

```
                    credit=course[i].Credits+credit;
                    System.out.println("You have successfully registered the
                    courses");

            }
            else
                    System.out.println("The registration
                     of "+course[i].name+"is failed, Please try other courses");

}

)
}
```