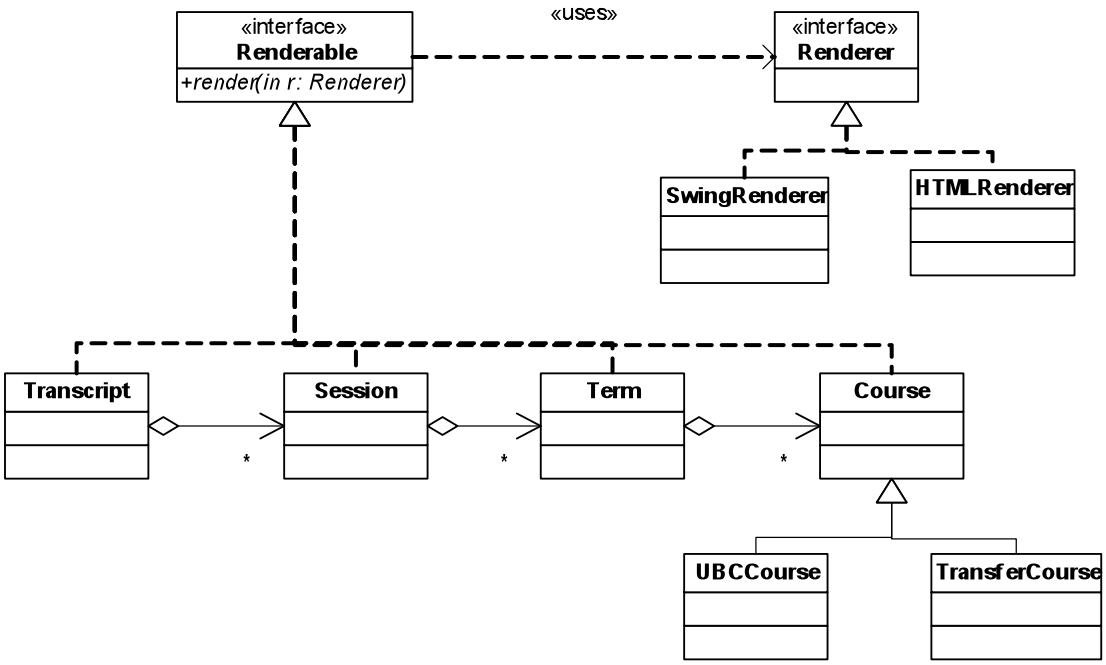
CPSC 211

SOLUTIONS to MIDTERM PRACTICE EXERCISES

Software Design

1. Although their may be other ways to design this problem, the following design is probably the simplest one.



2.
a)

	precondition	postcondition
`addItem`	*stronger*	*same*
`computeBill`	*same*	*stronger*
`checkOut`	*stronger*	*stronger*

b) `DeliveredGroceryOrder` is not a proper subtype of `GroceryOrder` according to the LSP because the preconditions on `addItem` and `checkout` are stronger. For the LSP to hold, preconditions in the subclass must be weaker (or the same) and postconditions must be stronger (or the same).

Exceptions

1. Output is:

```
catchit caught an exception
catchit caught an exception
In throwit a is 10
```

Software Testing

1.

a.
```
amount > 0
amount <= 0
```

b. Some sample test cases (others were possible):

```
1) theRow=25, theSeat = 50;    typical
2) theRow=1, theSeat = 1;      boundary
3) theRow=50, theSeat = 10;    boundary
4) theRow=49, theSeat = 1;     boundary
```

Java Collections etc.

1.

```
public static <E> void deleteAll(List<E> list,  E obj ){
   Iterator<E> itr = list.iterator();

   while ( itr.hasNext() ) {
      if ( obj.equals( itr.next() )
         itr.remove();
   }
}
```

This operation take O(n) time for LinkedList and $O(n^2)$ for an ArrayList. What if we don't use iterator and write:

```
public static <E> void deleteAll(List<E> list,  E obj ){

   for ( E cur : list ) {
      if ( obj.equals( cur ) )
         list.remove(cur);
   }
}
```

Is this a correct code?

2.

```
public static <E> List<Integer> getIndices(List<E> lst,  E obj) {
   List<Integer> result = new ArrayList<Integer>();

   ListIterator<E> itr = lst.listIterator();

   while ( itr.hasNext() ) {
      if ( obj.equals( itr.next() ) )
         result.add(itr.previousIndex());
   }
   return result;
}
```

This operation take O(n) time.

3.

```
public static <E> List<E> subst(List<E> list,E old, E new) {

   List<E> newlist = new ArrayList<E>();

   Iterator<E> itr = list.iterator();
   while ( itr.hasNext() ) {
      E nextElt = itr.next();
      if (nextElt.equals(old))
          newlist.add(new);
      else
          newlist.add( nextElt );
   }
   return newlist;
}
```

4.

```
public boolean equals( Object o )  {

   if ( o == null )
       return false;
   if ( getClass() != o.getClass() )
       return false;

   Dog d = (Dog) o ;
   return breed.equals(d.breed) && name.equals(d.name)
          && gender.equals(d.gender);
}
```