Background:
For this project we will be building from the CD and track classes you have been working with in lab. We will be modeling a 6 disk CD player. The player will allow us to insert CDs, play one track at a time, or program it to play as many tracks from as many different CDs as we like.

Details:
The interface for this project is posted at: http://courses.cs.vt.edu/~cs1705/Spring05/projects/p2/doc/. You will want to read through it all and make sure you understand what it is asking you to do.

There are a few changes with the classes you have already created. Most notably is the addition of the Time object in the Track class. Additionally the CD is now using the notion of time as its capacity measure as opposed to a track limit restriction. The constructor for the CD will receive as a parameter a time object that will indicate the amount of time that the CD can hold. The CD will also support a method to indicate how many tracks are currently on the CD.

The track class has a changed constructor to accept a time object as well. This parameter indicates the running time for the track. The track has an additional method to return the running time of the track. This is an integer and should probably be the running time of the track in seconds.

The two new classes are Time and CDPlayer. The time class is a class that will represent elapsed time. Think of it more as a timer then time like clock time. The two constructors are listed. The default constructor should set the time represented to 0. The parameterized constructor will store the numbers that it was given. I suggest, but am not requiring, that you think of a way to store the internally the time in one field. You can successfully complete this with three fields as well. The internals of the time object are up to you. The other methods for this object allow for resetting of the stored time, getting the currently stored value as an integer that is the total seconds stored by the object and a toString method that returns a nicely formatted string. This method will pad the minutes and seconds appropriately to make sure the times are always in the form h:mm:ss. A point of interest is if the parameterized constructor is passed values that would make the time round up, then you need to account for this. For example, if the hours = 1, the minutes = 72 and the seconds = 123, then you would need to add 2 to the minutes and then add 1 to the hours. The seconds and minutes will always be stored between 0 and 59. The hours can be any integer 0 or larger.

The final class is the CDPlayer class. This class will model a 6 disk CD player. You will store the CDs internally in an array. This is a requirement. You may not use an ArrayList. The class has three methods. The insertCD method accepts a CD parameter and will allow you put a CD in the player. The other two methods simulate playing tracks. The playTrack method will accept two integer parameters indicating which CD and which track to play. The CDs and tracks both start number at 1. You will have to
A 6 Disk CD Player

account for this in your CDPlayer logic. If the given CD and track numbers are valid you will return the information about the track, the title and running time. (Hey, we already have a track method that does that!!) The other method playTracks accepts two ArrayList objects that are the CD numbers and the track numbers to play. They should be of the same size or you return an empty string. If they are the same size, then you start making a (giant) string of all the track information. You will tell me the track playing order starting with 1 and then the track information. If either the CD number or the track number is invalid, then you still give me the track number but instead of the track information you say “Invalid Selection”.

Grading:
Grading will be done on Web-Cat and therefore you need to write test classes for all of your classes. The easiest way is to write a class that will test each class individually and make sure that it works. You will also have to make sure your style is correct.

Make sure you follow all of the interfaces closely. You will need to match the names of the methods and the type and order for the parameters.

Assistance:
If you need help with this project, do not hesitate to send me an email, come to my office, come see a TA. Get help early and get it often as needed. This is a learning experience and I want you to be successful.

Honor Code:
Since this is a project and not a lab, you are not allowed to get unauthorized assistance. That means not talking to other people who are not either the instructor or TAs about how to complete this assignment. If you would like to discuss general questions with others that is ok, but when you start talking about how to actually write the code the line has been crossed.

Additionally, make sure that the following comment is included in your code at the top of the file:

```
// On my honor:
//
// - I have not discussed the C++ language code in my program with
// any other student, either modified or unmodified.
// - I have not designed this program in such a way as to defeat or
// interfere with the normal operation of the Curator System.
```
A 6 Disk CD Player

//
// <Student Name>