

# Introduction to Computers and Engineering Problem Solving 1.00 / 1.001 Fall 2005

---

## Problem Set 3

Due: 11AM, Friday September 30, 2005

### TIVO 1 [100 points]

#### Introduction

The TIVO system allows viewers to schedule and record hours of their favorite television programs digitally. Eventually, we will simulate several components of this system. In this problem set, we will develop classes that may be used in later homework assignments.

There are several types of television programs. To keep things simple, we will assume there are four types: movies, news, series and specials. Each type of program has certain attributes. For example, some movies are comedic dramas while some news items show national and international business news. Additionally, each class holds information specific to the television program. For example, since "Friends" is a `Series`, each episode has an episode and season number.

All data members have been included in the source code we have given you but you must determine which data members should be `static`, `final`, `public` or `private`. You must also create the constructor for each class, specify the correct parameters of the constructor, and code appropriate `get()` and `set()` methods.

Since all programs have a duration, you are required to create a `Duration` class. This class has a constructor which takes integer values for `hours`, `minutes` and `seconds`, and stores these values. The class has `get()` and `set()` methods, as well as methods which calculate the total number of seconds, minutes or hours represented by the `Duration` object.

Programs are stored in a `DailySchedule`, in a two-dimensional `Object` array. The rows of the array represent the channels named in the source code, such that row 0 of the array of programs corresponds to location 0 in the array of channel names. The columns of the `Object` array represent 48 thirty-minute time slots. The `DailySchedule` also stores the date for which the schedule is valid as a `java.util.Calendar` object. The `DailySchedule` class needs a constructor, `get()` and `set()` methods, methods which convert the time slots and date to `String` representations and a `print()` method, which prints the entire schedule in a tabular format. For the `print()` method, you may use the abbreviations of the channel names.

The `ProblemSet3` class contains a `getDailySchedules()` method that generates an array of `DailySchedule` objects. The length of this array is specified by you. You must generate and print one `DailySchedule`, showing the schedule's date and showing all programs, channels and times.

## Assignment

Create or modify the `Movie`, `News`, `Series`, `Special`, `Duration`, `DailySchedule` and `ProblemSet3` classes provided in a zip in the assignments section, as follows:

`Movie`, `News`, `Series`, `Special`

- Determine the access modifiers for all data members (`public/private`). Also determine whether data members are `static` and/or `final`.
- Create constructors with values for the appropriate data members as arguments.
- Create `get()` and `set()` methods for the appropriate data members.

`Duration`

- Create the `Duration` class. This class has data members for hours, minutes and seconds.
- Create `get()` and `set()` methods for the data members.
- Write a `getTotalInHours()` method which converts the total time represented by the `Duration` object to a `double` value representing hours. For example, 1 hour 30 minutes would become 1.5 hours. Do the same for `getTotalInMinutes()` and `getTotalInSeconds()` methods.
- Write a `toString()` method that creates a `String` representation of the values represented by the `Duration` object.

`DailySchedule`

- Create a constructor which takes a `Calendar` object (date) and an `Object` array (programs) as arguments.
- Generate `get()` and `set()` methods for the appropriate data members.
- Write a static `dateToString()` method which converts a date (`Calendar`) to a `String`.
- Write a static `channelNumberToName()` method which converts the row number representing the channel into the channel name.
- Write a static `timeSlotToTime()` method which converts a time slot (0 through 47) into a time (`String`). This method should allow either a 12 hour or 24 hour format.
- Write a `print()` method which takes no arguments and returns nothing but prints a tabular representation of the `DailySchedule` object. Rows represent channels; columns represent times. The date of the schedule must be printed prominently. To help you, a `formatTo20Chars()` method has been provided. This method takes a `String` as an argument and returns a `String` that is 20 characters long.

`ProblemSet3`

- Generate and print a `DailySchedule` in the `main()` method.

## Turn In

- Turn in **electronic** copies of **all source code** (.java files). No printed copies are required.
- Place a **comment** with your full name, MIT server username, tutorial section, TA's name, and assignment number at the beginning of all .java files in your solution.
- Remember to **comment your code**. Points will be taken off for insufficient comments.
- Place all of the files in your solution into a **single zip file**. Submit this single zip file under the appropriate section and problem set number.
- Your solution is **due at 11AM**. Your uploaded files should have a time stamp of no later than 11AM on the due date.
- **Do not** turn in compiled byte code (.class files) or backup source code (.java~ files).

## Penalties

- **30% off** If you turn in your problem set **after 11AM on Friday** but **before 11AM on the following Monday**.
- **No Credit** If you turn in your problem set **after 11AM on the following Monday**.