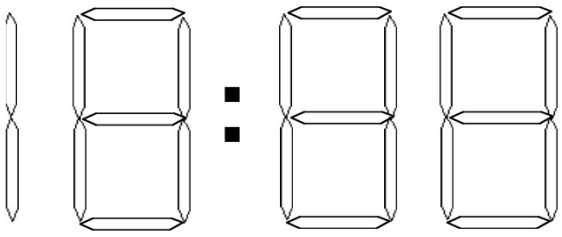
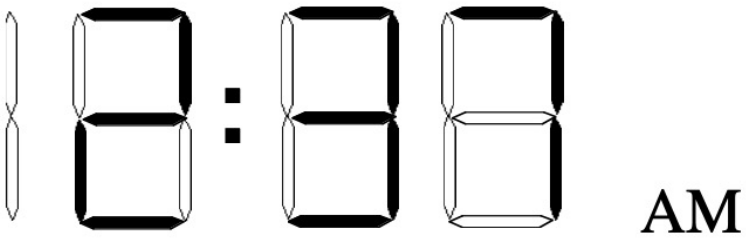
Lab09: List/Collections Lab — Due: Feb 5, 2019 at 3:30pm

The goal of this lab assignment is for you to master the various syntax and functions associated with lists. There will be two programs you are expected to code to successfully complete this lab.

1. You will write a program called Lab09_DigitalClock.py that will create a GUI (pseudo) digital clock. This clock should have the following format:



Notice that such a clock has a number of "cells" that are either filled or not depending on the numerical values contained. For example, 2:37 in the morning would show as:



You will display the time in a simple window that will contain a Text Entry box (for a user to enter the time) as well as a "Change" button once that has been entered. There should also be five other buttons: "+ Minute", "- Minute", "+ Hour", and "- Hour" to add and subtract minutes and hours from the time and a Quit button. Note that all entered times in the text box will be assumed to be AM.

When your window opens, your clock will show midnight (12:00 am). Users can click the minute and hours buttons to adjust or manually enter a new time in the text entry box. Your job is to properly fill the cells so that the time always displays correctly, including AM/PM. (You can get afternoon times using the buttons that add minutes or hours.)

- For full credit, you need to follow the structure of the dice example discussed in class and in the book. In other words, you should employ a table-drive approach (as discussed on pg 382). Your program will not be considered "successful" if you just have endless setFill commands on each individual cell.
- The actual shape of the cells is up to you as long as there are seven cells per number place. You are welcome to use the full seven to represent the leading one as well. Note that the outlines of the cells do not need to be visible when not filled – I showed them above just as an aid for understanding.

Hint: I highly recommend a Digit class somewhere that contains seven cells.

2. You will write a program called Lab09_TeamRanking.py that will read an input file containing the results of all games played in a particular sports' league season and generate a league ranking at the user's request. (This program can either be text-based or using a GUI.) You should first ask the user for the name of the input file which will have the following format:

```
<League Name>
<TeamName1>
<TeamName2>
.
.
<TeamNameN>
<Empty Line>
<Home Team><Space><HomeTeamScore> vs <AwayTeam><Space><AwayTeamScore>
<Home Team><Space><HomeTeamScore> vs <AwayTeam><Space><AwayTeamScore>
etc.
```

Then your program should ask the user how they want to rank the teams by wins, draws, or losses AND the trend (least to greatest or vice versa). If there are ties in the ranking, then teams sharing the same value should be listed in alphabetical order.

For example, an input file might look like:

```
Under-6 Girls Soccer League
Red Cats
Blue Dogs
Yellow Birds
Green Fish

Red Cats 2 vs Blue Dogs 1
Yellow Birds 3 vs Green Fish 3
Blue Dogs 0 vs Yellow Birds 1
Green Fish 4 vs Red Cats 2
Yellow Birds 2 vs Red Cats 2
Blue Dogs 1 vs Green Fish 1
```

If the user wants the team ranking by wins from greatest to least, the program should display the following:

```
Under-6 Girls Soccer League Ranking by Most Wins

Team           W     D     L
----           -     -     -
Green Fish     1     2     0
Red Cats       1     1     1
Yellow Birds   1     2     0
Blue Dogs      0     1     2
```

Hint: I highly recommend you write a Team class for this program.

You should submit at least three modules for this lab (Lab09_DigitalClock.py, Lab09_TeamRanking.py, and Button.py) but you are welcome to add more, especially if you have supporting classes (please name additional modules using the format "Lab09_*moduleName*.py").

This lab is worth 10 points and the grading will be as follows: 4 points for successful execution of each program, 1 point for adhering to the 100 line limit on functions, and 1 point for thorough comments.