The purpose of this lab is to give you practice working with format strings.

## Your Mission

Grab a copy of `lab8.py` from Blackboard: you will need to make a series of modifications to this file. The unmodified version of the file is included on the next page.

This file contains a series of assignment statements involving string formatting, such as the following:

```
answers[0]  = "#{3}#".format(one,two,three,four)
```

Each of these assignment statements is preceded by a comment that explains how the data should be formatted. Your task is to modify the slot descriptions as necessary to accomplish that formatting objective: in many cases, you'll need to change the index in addition to adding formatting directives.

In each case, the **only modifications you should make** are in the ***slot descriptions***; in the example above, the slot description is `{3}`. ***Do not change the parameters to `format` or anything else outside of the slot descriptions.***

**Verifying your answers:** The file has been set up so that you can simply run the script, and answers will be printed out to the screen. I encourage you to test things out as you go.

## What and How to Submit

You do not need to submit this lab through Blackboard.

**Instead:** Once you've verified your answers, print out your modified `lab8.py` and hand it in. (Make sure you've included your name and version of the course.)

```
# <Your name here: >
# CPS 300/500  (please indicate)

# create a 12-element list to contain our answers
answers = [1]*12

four = ""
one = 1234.56789
two = 268
three = "october"

# display one in a field of width 15, left justified
answers[0]  = "#{3}#".format(one,two,three,four)


# display one in a field of width 15, centered
answers[1]  = "#{3}#".format(one,two,three,four)


# display one with 20 significant digits
answers[2]  = "20 sig digs: {3}".format(one,two,three,four)


# display one with 4 decimal places
answers[3]  = "4 dec places: {3}".format(one,two,three,four)


# display both two and three (in that order),
#     in fields of width 10 that are right justified
answers[4]  = "#{3}#{3}#".format(one,two,three,four)


# display both three and two (in that order),
#     in fields of width 5 that are left justified
answers[5]  = "#{3}#{3}#".format(one,two,three,four)


# display two in a field of width 9, centered, with all blank
#     spaces replaced by periods
#  Result should be:     ...268...
answers[6]  = "{3}".format(one,two,three,four)


# display one in a field of width 14, right justified,
#     with all blank spaces replaced by zeros
answers[7]  = "{3}".format(one,two,three,four)


# set up the next four so that
#     (1) the decimal points line up with each other
#     (2) 1 decimal place is shown for answers[8]
#     (3) 4 decimal places are shown for answers[9]
#     (4) 3 decimal places are shown for the final two
#
# You'll have to play with field widths to line things up.

answers[8]  = "{0}".format(528.7568)
answers[9]  = "{0}".format(-32.17)
answers[10] = "{0}".format(1.357908642)
answers[11] = "{0}".format(16326.4)

############################################################
# display all the answers
for i in range(12):
    print ("answers[{0}]:\n {1}".format(i,answers[i]))
    print ()
```