EEN118 LAB FIVE

The purpose of this lab is to give you more practice with nice clean structured designs. Make absolutely sure your programs are very clear and obviously correct even to a reader who doesn't know what your design plan was. Grading will be on this basis.

The first two parts should be very fast. Remember the % operator. If you find that parts 1 or 2 are taking more than just a few minutes to work out, ask the Lab Guy for a hint. You have probably just not realised what an easy thing to do it really is.

1.   *What Time Is It?*

The library contains functions called `get_clock_time` and `get_calendar_date`. They both take no parameters and return an integer as their results. The first tells you the time of day, hours, minutes, and seconds, squeezed into six digits: at exactly 3:30 p.m. the value would be 153000. The second tells you the date, year, month, and day, squeezed into eight digits: on the 18th of September 2009 the value would be 20090918.

Write a simple program that obtains the current year, month, day, hour, and minute as separate integers. After a little calculation, your program should contain these five lines or something very much like them:

```
print("  year: ");  print(year);   newline();
print(" month: ");  print(month);  newline();
print("   day: ");  print(day);    newline();
print("  hour: ");  print(hour);   newline();
print("minute: ");  print(minute); newline();
```

when run, it should produce output looking something like this:

```
  year: 2009
 month: 9
   day: 18
  hour: 15
minute: 30
```

but with the correct numbers of course.

Consider for a moment what very rare problem could possibly occur if you start to run your program just a tiny fraction of a second before midnight. What could your program do guard against that problem?

2.   *The Monroe Doctrine.*

Who uses the continental clock? Everyone knows hours are supposed to range from 1 to 12, not zero to 23. Modify your program so that the hour is reduced to the proper 1 to 12 range, but make it also note whether the time is "`a.m.`" or "`p.m.`".

```
     year: 2009
    month: 9
      day: 18
     hour: 3 p.m.
   minute: 30
```

(The Monroe doctrine was the early 18th Century U.S. policy, introduced by President James Monroe, intending to remove the European influence from North America. I'm sure you already knew that.)

3. *A Clock Face.*

Create a window slightly taller than it is wide, and draw a large circle nicely positioned within it. Make twelve marks regularly spaced around it near the edge, so that it can act as a clock face.

   You don't need to duplicate this sample, it is just there to make the plan clear.
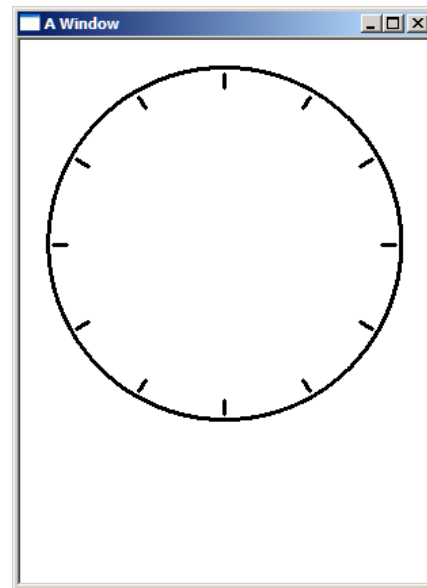
4. *A Whole Clock*

Making use of your programming for parts 1 and 2, draw hands on your clock at the correct positions for the current time of day. A little trial-and-error experimentation may be needed to get things just right.
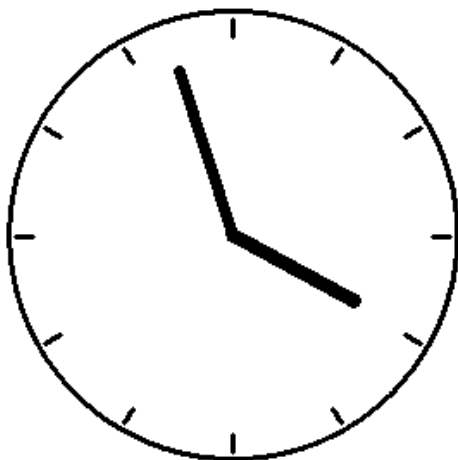
A hint towards correctness:
   *The time is 3:57, so the hour is 3, but where should the hour hand be pointing?*

5. *Animate Your Clock.*

Make your clock actually run, so that it keeps updating itself to show the correct time continuously for as long as you let the program run. You will find testing much easier if you add in a second hand. It can be really boring watching the minute hand for long enough to know that it moves correctly.

6. *A Complete Product.*

This is why the window isn't square.

Use the space under the clock to display the time and date in a nice human-friendly way. Don't go overboard with detail, but try to make it look nice.
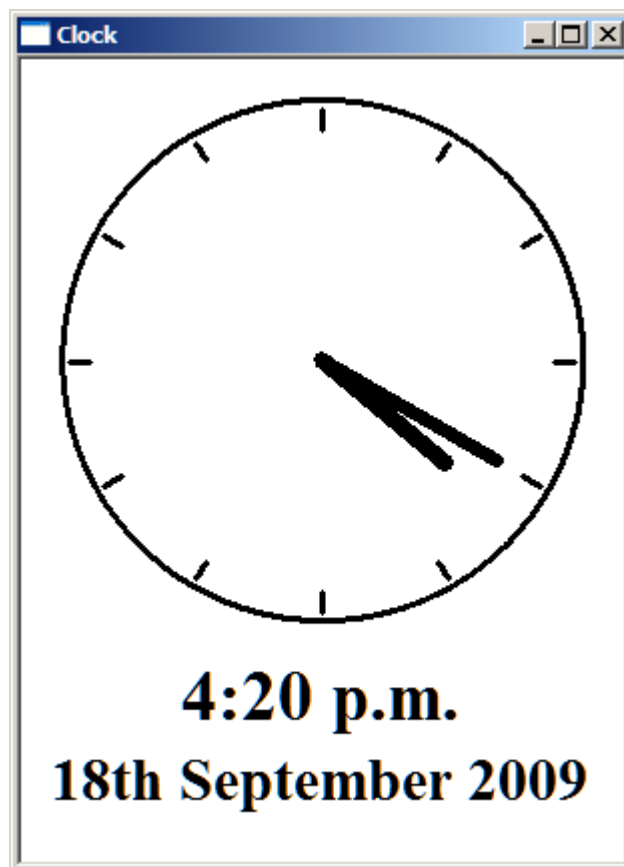
You have already seen the `write_string` function, but you may not have noticed that it doesn't only display strings, it will happily render `int`s and `double`s: whatever you throw at it. The date was produced by four separate calls to write_string: one for the 18, one for the "th", one for the "September", and one for the 2009.

What are you going to do to ensure that 1405 is displayed as 2:05, and not 2:5?