EAS 199A Direct Solution Format

Overview

The *Direct Solution Format* is a terse form of documenting solutions to homework assignments, and it is used when the longer form, *Standard Engineering Format* (described in a companion document) does not apply. The *Direct Solution Format* may be specified for problems involve direct evaluation of a formula. The *Direction Solution Format* may be specified for problems that require you to build circuits, assemble hardware, or write programs. The appropriate format of the problem will be specified for each assignment. You should develop your own judgment on the appropriate solution format, but do not use that judgment to override an explicit specification given on the assignment

The nature of the direct solution will vary with each type of problem. The unifying idea is that the direct solution only requires immediate responses posed in the problem statement. In other words, when the *Direct Solution Format* is appropriate, there is no need to use the Given, Find, Solution/Analysis and Discussion sections from the *Standard Engineering Format*. For problems where the *Direct Solution Format* is appropriate, there will be often be explicit instructions in the problem statement about what to include in the solution.

It may seem like requiring neat solutions with lots of details is just busy work. That is not true. Engineers do not work in isolation. We need to document our work to enhance communication with other members of our team. Often we will need to revisit work that we performed earlier. In those cases, good documentation will save us the effort of having to re-do work that was poorly documented in the first place.

General Requirements

As with the *Standard Engineering Format*, the following general requirements apply to the solutions with the *Direct Solution Format*.

1. Solutions should be NEAT. Do not cross out text or formulas except to indicate an algebraic cancelation. Make sure erasures are complete, i.e., that the previous text does not show through.
2. Use engineering paper (yellow, green or white). Plain white paper should not be used with handwritten solutions.
3. NEVER turn in solutions on paper that is ripped out of a notebook.
4. Staple pages of an assignment together. NEVER turn in an assignment on loose sheets of paper. Folding over the corner of a stack of loose sheets is unacceptable.
5. Present the problems in the order they are assigned. If you run out of time and skip a problem, you can indicate that by including the problem number and “skipped” or “Could not solve” or some other note to the grader. The note is not required, but it saves the grader from scanning through your solution packet to look for your missing solutions.
6. Number all pages in the upper right hand corner of the paper. For a 5 page assignment, the first page would be numbered 1/5, the second page would be numbered 2/5, and so on.
7. Use of MathCAD, Excel or MATLAB to complete all or part of a homework assignment is encouraged. Some assignments will explicitly require use of software..
8. Use the proper units throughout the solution.
9. If a solution fills one half page or more, begin the solution to the next problem on a new page.
10. If the solution to more than one problem is included on a single sheet:
    - Draw a horizontal line in the space between solutions. The line should at least be as long as one third of the width of the sheet of paper.
    - Do not crowd the solutions. Leave at least 1 cm blank space between horizontal line at the end of one solution and the start of the next solution.
11. Long problems can span multiple pages. Do not crowd a solution so that it will fit on a single page.

Code Documentation

Solutions involving computer code introduce some addition considerations. Integrating code and output in a handwritten document often causes enough complications that it is better to paste the code and output into a word-processor.

- Use mono-spaced font for code listings. Mono-spaced font allows the use of spaces to create vertical alignment within the code.
- Include concise comment statements. Code should always be commented. The comment statements in the code should be brief, and not take up excess space. If more extensive explanation of the code is appropriate, e.g. documentation of complex formulas, include additional notes in the solution document. There is no fixed rule about how much in-code documentation is too much.
- If the solution is handwritten, either include the code on a separate sheet, or if the code is short, cut out the code from a printed sheet and paste it securely on the handwritten solution. Do not include loose half-sheets of paper.
- Text and numerical values printed to the Serial Monitor, should be printed in mono-spaced font. If the solution is handwritten, present the output in the same format as code, which is described in the preceding bullet.

Example

Assignment:

1. Download and install the latest version of the Arduino IDE on your laptop.
2. Build a circuit and write the Arduino code to blink an LED on/off every 2.5 seconds. In your solution, include a diagram of the circuit and the Arduino code.
3. How many valence electrons are in a 1cm cube of copper? The following data may be useful. The atomic mass of copper is 63.55 g/mol. The density of pure copper is 8.94 g/cm^3. Avogadro's number is 6.022×10^{23} atoms/mol.

Solution:

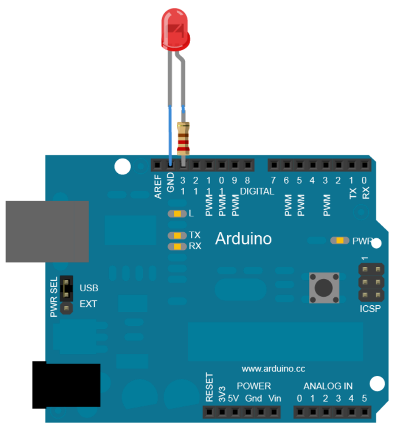
1. I downloaded and installed version 023 of the Arduino IDE from www.arduino.cc
2. I used the blink.pde program available from the examples menu:

   File → Examples → Basics → Blink

   I inserted the positive terminal of the LED to the socket for digital output 13. I connected the ground terminal of the LED to a 330Ω resistor, and inserted the other end of the resistor into the GND socket adjacent to the socket for digital output 13. The circuit is represented by the sketch on the following page, which was copied from the Arduino web site (http://www.arduino.cc/en/Tutorial/Blink).

   The example code needed to be modified to extend the blink and pause intervals to 2.5 seconds or 2500 milliseconds. The final version of the code is listed at the top of the next page. The comments were changed to identify the source of the code, and to explain the duration of the blink interval.

   Compiling the code and downloading it to the Adruino is straightforward. The code runs and the LED blinks as required by the problem statement.



Schematic of the blink circuit from http://www.arduino.cc/en/Tutorial/Blink

```
/*  File: blink25.pde

  Turn an LED on for 2.5 seconds, then off for 2.5 seconds.
  Repeat indefinitely.

  Code is a slightly modified version of the blink.pde code
  available from http://www.arduino.cc/en/Tutorial/Blink.

 */

void setup() {
  pinMode(13, OUTPUT);      // Configure digital pin 13 as an output.
}

void loop() {
  digitalWrite(13, HIGH);   // set the LED on
  delay(2500);              // wait for 2.5 seconds
  digitalWrite(13, LOW);    // set the LED off
  delay(2500);              // wait for 2.5 seconds
}
```

---

3. First compute the number of atoms, N

$$N = 1\,\text{cm}^3 \times 8.93\frac{\text{gm}}{\text{cm}^3} \times \frac{1\,\text{mol}}{63.55\ \text{g}} \times \frac{6.022\times 10^{23}\,\text{atoms}}{\text{mol}} = 8.5\times 10^{22}\ \text{atoms}$$

Since each copper atom has one valence electron, there are 8.5 × 10^{22} valence electrons in a 1 cm cube of copper.

Answer: 8.5 × 10^{22} per cm^3 of copper.
