ECE101: Exploring Digital Information Technologies **Fall 2023**
Week 1 Exercises (with solutions in red) **based on L1,L2; tested on Exam I**

Information and Computation

These exercises are intended to help you master and remember the material discussed in lectures and explored in labs. In future semesters, we may make some or all of these exercises required, but for now they remain optional. We suggest that you do them as we go over the material, but you may also want to use them to review concepts before the exam. You may notice that all of the questions here are based on Lecture 1 (L1), as L2 was a historical overview of technologies and not meant for you to remember in detail. Future weeks' exercises will include problem based on the material in both lectures.

Rather than using this version directly, we suggest that you use the version without solutions to solve the problems before looking at the answers. Many studies have shown that people often trick themselves into believing that they know how to solve a problem if they are presented with the answer before they try to solve the problem themselves.

2^0	1
2^1	2
2^2	4
2^3	8
2^4	16
2^5	32
2^6	64
2^7	128
2^8	256
2^9	512

1. [L1] Recall that any information, such as a choice among a group of things, can be represented using a sequence of bits (a binary number). How many bits are **needed** to represent something from each of the following? *For your convenience, a table of powers of 2 is given to the right.*
   A. A sandwich from the "Originals" list on the Jimmy John's menu: The Pepe, Big John, Totally Tuna, Turkey Tom, Vito, or The Veggie.
   B. One of the fifty states in the United States.
   C. One of the days in the month of September in the year 2022.
   D. A current ECE Illinois senior—there are approximately 400.

   A. 6 sandwiches, so 3 bits.
   B. 50 states, so 6 bits.
   C. 30 days, so 5 bits.
   D. 400 people, so 9 bits.

2. [L1] Let's say that you want to record the temperature inside and outside of your apartment window over the course of a day. Your thermometers provide you with a 2 Byte (16-bit) sample—say degrees in F or C, whichever you prefer. For simplicity, let's say that your system produces a sample 50 times each second. How many total Bytes do you need to record the day's temperatures? Show your work—*writing the answer as a product is sufficient*.

   2 B × 50 samples / second × 60 seconds / minute × 60 minutes / hour × 24 hours / day × 2 thermometers (17,280,000 Bytes = 16.5 MB for the raw data; in practice, one could compress the data significantly)

3. [L1] Why do we say that the prefixes used in computing technology—kilo, Mega, Giga, and so forth—are not quite the metric system?

   Computing prefixes often (but not always, just to make things more confusing!) refer to powers of 2 rather than the powers of 10 used in the metric system. For example, a kilogram is 1,000 grams, but a kilobyte is 1,024 Bytes. The difference (2.4%) usually doesn't matter much but gets bigger with bigger units (more than 7% with Giga).

4. [L1] Let's say that you have a set of things and want to record information about each thing. A Huffman code uses the frequency of the possible answers to compress the information. For example, imagine that we ask each of 100 students to choose an ECE101 T-shirt color: black, white, red, or yellow. Let's say that 70% of students choose black, 15% choose white, 5% choose red, and 10% choose yellow. A Huffman code then assigns bit patterns as follows: black is just 0, white is 10, yellow is 110, and red is 111.
   A. How many bits in total are needed to store the students' answers if two bits are used for each answer?
      There are 4 color options, so we will need 2 bits × 100 students for a total of 200 bits.
   B. Assuming that the percentages are correct, how many bits are needed to store the students' answers using the Huffman encoding?
      70 of the students (70% of 100) need 1 bit (black encoded as 0), 15 need 2 bits (white), and the remaining 15 need 3 bits (yellow and red), for a total of $70 \times 1 + 15 \times 2 + 15 \times 3 = 145$ bits. That's compression!
   C. What happens if our expectations about frequency are wrong? Assume now that all four colors are equally popular, so we have 25 of each of the four colors. How many bits are needed to record the students' answers using the Huffman encoding in this case?
      25 of the students need 1 bit (black encoded as 0), 25 need 2 bits (white), and the remaining 50 need 3 bits (yellow and red), for a total of $25 \times 1 + 25 \times 2 + 50 \times 3 = 225$ bits, which is more than is needed without "compression!" As mentioned in class, compression schemes leverage our knowledge of frequency distributions, but can produce worse results when our expectations are incorrect. In practice, it's often easy to decide not to use compression when that happens.

5. [L1] Recall the Church-Turing hypothesis discussed in lecture.
   A. In your own words, restate the hypothesis.
      The Church-Turing hypothesis claims that humans and computers are equally capable of computational tasks. If a computer can solve a problem, so can a human, and vice-versa, given enough time and memory. There ARE problems that cannot be solved by either.
   B. In lecture, we mentioned that problems like finding boats, trains, airplanes, or cats in photographs are often used to differentiate human users from "bots" (computer programs) over the Internet. Explain why that approach was effective, given the Church-Turing hypothesis. (It's less effective today, but still used because AI approaches require substantial computation power for speed.)
      We don't know how our brains work, and for decades were unable to program computers to perform basic image recognition tasks at any reasonable speed or accuracy. Believing that something is possible does not tell you how to do it, so humans' inability to program computers to solve a problem is not a contradiction to the Church-Turing hypothesis.
   C. A computer can find and sum the first 1,000 primes in milliseconds. Can you? If not, explain why, given the Church-Turing hypothesis.
      The Church-Turing hypothesis is only true given enough time and memory. Computing works at different speeds across computers (including our brains) and may be different for different problems—humans are really good at predicting trajectories, for example—that's why you can catch a ball—if you had to think about catching the ball, you would always fail (out of time!). So the fact that none of us is as fast as a computer for finding primes or adding is not a contradiction to the Church-Turing hypothesis.