1. After reading (the) DOET, I have realized how frustrating simple faucets can be. Many different types of faucets are used daily by at home, at school, and at work. They are used, among other things, to wash away dirt from hands or food from plates and as a source of drinking water. Because they are so prolific, they should be designed for anyone to use intuitively – with as little thought and planning as possible. My bathroom faucet has a few advantages; the hot and cold knobs are marked red and blue respectively, cold water is controlled by the knob on the right while hot is controlled by the knob on the left, and the volume and temperature of the water is adjustable. There are some aspects that could be changed; the water temperature reaches scalding quickly, the cold knob requires more turning for the same amount of flow, and my mental model consists of turning both knobs inward to ‘mix’ the temperatures while in reality both must be turned counter-clockwise.

The lock on my apartment door seems like a fairly simple mechanism. Like faucets, door locks are used almost everywhere – from apartments to offices to Siebel Center. Their primary purpose is to prevent people from entering a room or area. Designers of door locks should take into account the ease and quickness of changing between locked and unlocked states and the sturdiness of the locking mechanism. Above all, once the lock is properly engaged, our door is very sturdily locked. The handle/lever used to transition between locked and unlocked states is easy to distinguish since it is a different color than the rest of the door. The handle/lever is also easy for people of varying dexterities to turn. That said, there is no obvious feedback when the mechanism reaches its locked or unlocked state – a ‘click’ or something similar would be helpful. When attempting to unlock the door from the outside, there are no visual clues to the way the key should fit in the lock. In order for the door to be securely locked, it must be closed tightly; when the door is not closed completely, it may seem like it is locked when it is not actually.

The street crossing systems scattered around campus are some of the nicest I’ve seen anywhere. They need to be; stop/walk signs and buttons help remind people not to walk into the path of an oncoming vehicle as they go about their daily routines. These systems should be designed to be very visible – to stand out and convey the status of the light in any condition. The systems around campus use clear symbols of a person walking for walk and a palm outstretched for stop. These symbols are intuitive – even white and red colorings are used to further convey the message. The signs are located high enough off the ground to be visible, and oriented towards the waiting pedestrian. The buttons make a satisfying ‘click’ sound when pressed, which seems like appropriate feedback. The systems emit a regularly spaced beeping sound from time to time, but it isn’t immediately obvious whether this means it is ok to cross. It seems like this sound is for the benefit of the blind, and they do not have the visual symbols to associate with the sound. Other
than a lack of the systems at minor intersections, no other flaws are immediately obvious.

2. **Mike**

Describe 3 problems you often encounter when interacting with your computer system and/or applications.
1. Finding and installing DVD software necessary to watch a movie.
2. Norton Antivirus takes forever to scan, finds a problem then says it can’t fix it.
3. MSN Messenger starts up with the system but can’t be shut down / won’t go away.

How frustrating is it to encounter these problems?
Not frustrating enough not to wait out, but very annoying.

How much to these problems affect your ability to get work done?
Takes up a large portion of time that could otherwise be used to do work. The computer is always updating, always needs to be restarted. The Windows restart timer box is annoying.

What should be done to correct these problems?
There should be universal software standards. People don’t like monopolies, but they would make life easier. Firefox is nicer than IE, but many websites can’t use Firefox. Norton should be simpler, and just do what it’s supposed to.

**Stephanie**

Describe 3 problems you often encounter when interacting with your computer system and/or applications.
1. I sometimes have to restart my laptop because the wireless isn’t working. It works for everyone else, but I need to restart.
2. There are a lot of programs’ things that pop up reminding about security, renewals and stuff. I don’t have Norton because you have to pay for it.
3. I think there are issues with my CD-ROM and one of my USB ports. Sometimes CDs and flash drives are inserted and the computer acts like there’s nothing there.

How frustrating is it to encounter these problems?
Very!

How much to these problems affect your ability to get work done?
Restarting is a hassle… makes like 5 minutes get wasted. Everything else you just click out of or switch USB things.

What should be done to correct these problems?
Hire someone that can fix it.

3.
See attached source code and output file+spreadsheet for data.
**Power Law of Practice:** \( T_n = T_1 \cdot n^{(-\alpha)} \) where \( \alpha \) is the learning constant, \( n \) is the number of trials. \( n = 20. \)

\[
T_{20} = 7 \cdot 20^{(-\alpha)} \\
2 / 7 = 20^{(-\alpha)} \\
\alpha = \log_{20}(2 / 7) \\
\alpha = 0.418
\]

At first, it took a while to think of the order of the letters. The style was a quick hunt-and-peck with the favored right hand. Then (the program displays the typed text – knowledge in the world) it was easier to remember what to type. Finally, the left hand was used, with fingers pre-placed on the initial keys.

4. Error free behavior, user knows target, static menu, 12 items.

(a) how long will it take a user to select an item from the menu?

\[
MT = a + b \cdot ID \\
ID = \log_2(A / W + 1) \\
MT = a + b \cdot \log_2(A / W + 1)
\]

*Assume that a Microsoft Mouse is used as in lecture. So \( a = 548, b = 420. \)
\[
MT = 548 + 420 \cdot \log_2(A/ W + 1) // A = area in px, W = width in px (of item) \\
MT = 548 + 420 \cdot \log_2(H + 1) // H = height in px = A/W
\]

*Assume that the height of a menu item is 20px. Since there are 12 items, the average height travelled \( 12 \cdot 20 / 2 = 120 \)px

\[
MT_{avg} = 548 + 420 \cdot \log_2(120+1) = 3,453.92256 \text{ units of time (ms?)}
\]
(b) first 4 placed dynamically (8 static) – how long to select when prob target in d/s:
   a. percentages:
      i. 50/50: \(0.5(548 + 420 \cdot \lg(40+1)) + 0.5(548 + 420 \cdot \lg(80+1)) = 3,004.45442 \) units of time
      ii. 75/25: \(0.75(548 + 420 \cdot \lg(40+1)) + 0.25(548 + 420 \cdot \lg(80+1)) = 2,901.31313 \) units of time
      iii. 90/10: \(0.90(548 + 420 \cdot \lg(40+1)) + 0.10(548 + 420 \cdot \lg(80+1)) = 2,839.42836 \) units of time
   b. minimum choice time? When 100% of selections are in the dynamic area: \(1(548 + 420 \cdot \lg(40+1)) + 0(548 + 420 \cdot \lg(80+1)) = 2,798.17184 \) units of time
   c. prob split causing max choice time? When the new list is equivalent to the old (prob. proportional to # of entries). Then it has the original average MT.

(c) what are limitations of applying Hick's Law to real-world performance tasks?

Determining the constants for each application of the law. This changes with every user and device.