Homework 2

1. Thinking about user interface design

Minivan: A minivan is a family vehicle often associated with “soccer moms” -- i.e. parents who are active in their many children's activities. The main purpose of a minivan is often for the transportation of school-age kids. Everything from the sliding door to the trunk of a minivan is designed with this purpose in mind. I'm examining the design of my family minivan:

Strengths:
- The sliding door allows for a very wide opening while minimizing the horizontal space taken up by the open door. This makes entrance easy since people can climb into the car rather than just slide into the seat.
- Through a relatively painless process, the back seat folds up, allowing more space. Conspicuous loops that look like they should be pulled make folding the seat intuitive.
- Sliding doors on both sides. It annoyed me when older minivans would only have a door on one side. It's just inconvenient to have to force everyone to one side.

Weaknesses:
- The door locks when pulled all the way back (in order to prevent inadvertently sliding shut and crushing fingers) and most people trying to close it again get confused and ask for directions to close it (directions: pull really really hard). Maybe lighten up how much force is required to pull it shut, especially since kids will probably be doing the closing.
- Back windows don't roll down, they just pop open like half an inch. With active kids, the first thing they want to do is roll down the window of the seat next to them, but can't, most likely because of the car wheel. This is a hard problem to solve. You could maybe compromise by keeping the window one that pops open half an inch but is controlled by a rotating handle.
- Child safety locks. It's hard to tell if they're activated or not. I think the only way is by opening the door and looking at its side and examining the switch. This is kind of a pain because the driver has to get out of their seat and open the door just to check. It should be an on/off switch somewhere near the driver's seat/dashboard, similar to a normal lock.

Dorm room bed: Beds made for dorm rooms need to take a lot into account, probably the most important of which is space. They also need to be comfortable for sleeping, since that's their primary function. I will add a second function, which is as a chair since the sitting space in dorm rooms is limited. Examining my dorm room's bed:

Strengths:
- Headboard. Being about 6 feet off the ground, it's nice to have at least one side preventing you from falling off. Plus it keeps light from the window out when I'm trying to sleep.
- There is enough space under the bed to store things like drawers and refrigerators. This is pretty space-efficient considering you're taking advantage of vertical space that would normally be wasted.
- The metal frame of the bed is thin so you can work wires around it without being hindered by bulkiness.

Weaknesses:
- Headboard blocks light. There is only one room light, and the headboard is right between it and my desk. Adding a second room light would help a lot, even if it's not that bright.
- The bed is so high up that it's impossible to study on it – it just makes you claustrophobic and uncomfortable. The bed could be lowered a bit.
- The ladder is made of thin metal bars, which are painful to climb on. In addition, there is a shelf which blocks how far your foot can go on the bar, which makes it hard to grip. It would be nice if the bars were more elliptical in shape (wider and flatter) to make stepping on them easier.

**Cell phone:** Cell phones are first and foremost for talking with people by phone. However, their small size, portability, software, and ability to vibrate and/or make loud noises make them useful for more than communicating. They become replacements for various tools including watches, alarm clocks, and flashlights. Examining my Motorola phone:

**Strengths:**
- The phone is really good at handling the case where you're talking with someone and you get an incoming call – it makes a beep, which makes you take your ear off the phone and look at the screen, which says you have another call and then gives you the option to switch to it or not. The person using the phone doesn't need any kind of preparation or instruction in order to take an incoming call in the middle of another one.
- Using the phone as a watch is easy. Pushing any button on the phone will display a screen with the time. You don't have to open the phone or anything.
- When I input a phone number and call it, and that phone number matches with an entry in my contacts list, it'll display the name of that contact as it dials. It's nice as a double-check, just so I know that the number I typed from memory is correct.

**Weaknesses:**
- I still have not figured out how to silence a call. Since this is a pretty common need, it should be intuitive. You should not have to guess which of the three buttons (if any) silences the call – there should be a switch whose purpose is clearly to silence the phone.
- The alarm clock feature of the cell phone has no way to save commonly used alarms. You can have an alarm repeat Daily, Weekdays, or Weekends, but for a college student, it's not uncommon to wake up at 7:30 a.m. MW, 9 a.m. TR, and 12 p.m. F. There should be a way to save commonly-used alarms or include more repeat schemes so that I don't have to input 7:30 on Monday, 9 on Tuesday, redo the 7:30 for Wednesday, then redo the 9 for Thursday, etc.
- Adding a contact is not intuitive. It actually requires only two clicks of the same button (Menu -> New Contact). However, I gravitate toward the “Contacts” on the right side, which instead brings up a list of contacts. I read the three options (Edit, View, Options) before realizing Add is not there. It is under “Options” even though adding a contact is not exactly an option. I think choosing “Contacts” should bring me to the menu that Menu does (this menu is also called Contacts). That way, I can easily decide from there to add a contact (the first thing on the list) or view my contacts list (the second thing). The “Menu” button is then freed up to go to a different place, like “Settings & Tools”.
2. Learning to listen to users

Many of the problems people talked about were related to operating systems. One person's complaints were all centered around Vista. His biggest issue was that erratic, unexplained, and just plain weird issues would spring up. For example, one day the Adobe Acrobat utilities panel in Office disappeared. After several uninstalls and reinstalls of the tools, he found out that the utilities had been moved to the printing menu. Other unexpected behavior included folder views – the person would switch from one folder view (small icons) to another (details). The next time the folder was opened, the view was back to small icons.

Another issue he had was that there were so many changes to Vista from XP that it created a learning curve. There were too many changes made to things that weren't broken, resulting in a lack of continuity. That probably describes the first problem too: lack of continuity. In a good interface, things should not change unexpectedly without explanation or the influence of the user, and if the user influenced it, it should stay that way unless they change it back. To fail to insure that this happens will end up being confusing and disorienting, with a big impact on productivity for the first few months.

The often unnecessary tweaks to the interface were probably caused by having the goal for Vista to be to break out of the similar series of Microsoft OS's of the past decade. However, care needs to be taken to assess beforehand whether features being left out or replaced are actually broken.

Another user complained about the lack of certain features on Macs that he finds especially useful for productivity, namely a button that deletes the character immediately after the cursor and a complete alt-tab function for switching between different windows. You can have four different Firefox windows open, but Firefox only appears once when alt-(cmd)-tab is used on a Mac. A separate key press actually switches between windows of a program, but it's not obvious. The result is that a lot more time is spent hunting for windows. There are perhaps several reasons for not including these features (aesthetics, not enough space on the keyboard, consistency with previous designs, copyright issues?), but it would be nice to have them. The delete button problem could be fixed by having “delete” remove the next character when a certain key is being pressed or something.

Other issues dealt with applications themselves. The second interviewee found the movable (and removable) toolbars in Microsoft Word more trouble than they were worth – he would accidentally move one or get rid of it and then have to spend time trying to get it back in place. The rationale behind this is reasonable: the drag-and-drop toolbars give an easy interface for customizing toolbars. However, some people (like this person) would prefer if the default setting for toolbars is locked. I think it would be good to have an icon on the right of a toolbar that locks it in place if you double-click it or something similar. That way, toolbars can remain unlocked by default but a user can easily lock it at any time.

Other issues were even higher-level: web interfaces. Having to navigate non-intuitive sites such as Compass was a source of frustration for a third interviewee. The loading times as well as the odd placement of information (discussion section numbers placed under grades, for example) made navigation of the website a chore. The speed of Compass and other annoyances (such as having to accept a certificate every time you enter it) are caused by Compass being built on Java. It's not a bad idea because of the capabilities a web application like Compass would need to have, but Compass does seem like it needs to have a redesign keeping in mind the things that Compass is commonly used for.
3. Power Law of Practice

Data:

<table>
<thead>
<tr>
<th>Trial</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>11.125</td>
<td>8.985</td>
<td>6.266</td>
<td>5.984</td>
<td>5.844</td>
<td>5.500</td>
<td>4.875</td>
<td>5.375</td>
<td>5.375</td>
<td>5.859</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trial</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Trial</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
<th>25</th>
<th>26</th>
<th>27</th>
<th>28</th>
<th>29</th>
<th>30</th>
</tr>
</thead>
</table>

Graph:

Learning Constant: \(0.189\)

This is rather low. The meaning of this low number is that there was not a huge difference between performance without practice and performance after a lot of practice. Some differences in times in later trials are almost as dramatic as the difference between practice and no practice.

Other Observations:

The typist first started out using just one finger and after a while started using multiple fingers to input characters. Throughout the trial a significant amount of time was taken up by deleting incorrect letters and double-checking input. While the time for actually inputting tended to drop, those times (deleting and checking) stayed pretty much the same.
4. Choice Reaction Time

(a) We need to find movement time in milliseconds (MT). There are 12 items. Assuming you start at the top of the list, you would have to move 6 units (a unit meaning the size of a button) on average. Let W be the size of a button. Using the equation and mouse parameters given in lecture:

\[ MT = 545 + 420 \times \log_{2}((6W/W) + 1) \]
\[ = 545 + 420 \times \log_{2}7 \]
\[ = 1724 \text{ ms} \]

(b) 50/50:
In the dynamic case, there are 4 items. Assuming you start at the top, you would have to move 2 units (=W) on average.

\[ MTD = 545 + 420 \times \log_{2}((2W/W) + 1) \]
\[ = 545 + 420 \times \log_{2}3 \]
\[ = 1211 \text{ ms} \]

In the static case, there are 8 items, located below the 4 dynamic items. Assuming you start at the top, you would have to move 8 units (=W) on average.

\[ MT_S = 545 + 420 \times \log_{2}((8W/W) + 1) \]
\[ = 545 + 420 \times \log_{2}9 \]
\[ = 1876 \text{ ms} \]

These two cases are equally likely to occur so we take the average:

\[ MT_{AVG} = (1211 + 1876)/2 \]
\[ = 1543 \text{ ms} \]

75/25:
We can reuse the times for MT_D and MT_S found above, but this time, we take a weighted average (weighted 75/25):

\[ MT_{AVG} = 1211 \times .75 + 1876 \times .25 \]
\[ = 1377 \text{ ms} \]

90/10:
Now we take an average weighted 90/10:

\[ MT_{AVG} = 1211 \times .9 + 1876 \times .1 \]
\[ = 1277 \text{ ms} \]

Minimum Choice Time:
The minimum choice time would be 1211 ms, which is just the time it takes to select from the dynamic part of the menu. This is assuming that you would only ever need to choose from the dynamic part (i.e. the dynamic/static probability is 100/0) – this would in effect make the menu four items instead of twelve.

Maximum Choice Time:
The probability split that would create the maximum choice time is 0/100. This would be even worse than having a normal 12-item menu – it would mean that you ALWAYS end up having to go to the lower two-thirds of the menu.
(c) Hick's Law applies to quick decisions from several options. It doesn't really apply to decisions that require active thinking. For example: measuring the time it takes for someone to pick one of several answers to a question. The thing that takes up most of the time in this case is NOT registering the presence and meaning of the options but considering and processing the question itself.