Managing Your Attention

The Problem of Distraction

Learning to code is *hard work* and I'm pretty sure that I'm a lot less exciting than your average YouTube star, so if I have to compete with them (or with your friends on WhatsApp, Snapchat, WeChat, or whatever) for your attention then I'll probably lose. And your computer and phone are *designed* to distract you because they are interested in *engagement*—they don't care that you should be engaging with your studies, they want you to pick them up and use them which is why you get so many notifications.

There is plenty of evidence to back this up:

- Distractions make learning harder
- Nearly half of students distracted by technology
- The effect of cellphones on attention and learning…
- Distractions make retaining info harder
- The interrupted learner: how distractions during live and video lectures influence learning outcomes

So my recommendations for studying online are:

- **Turn off as many notifications as possible**. On the Mac there is a 'Do not disturb' setting that you can enable in the Notifications section of your `System Preferences`. You can have it turned on by time, and also add the setting to your 'Notification Centre' (upper-right corner of your desktop).
- **Uninstall messaging apps that you cannot turn off**. If quitting them disables notifications then fine. If you can't disable notifications then I'd suggest uninstalling the messaging app entirely.
- **Block access to distracting web sites**. There are a number of tools that you can set up to block access to Facebook and other social media sites at set times of day.
- **Work out a schedule**. It's a lot easier to avoid distractions if you have a routine that enables you to say "OK, I will work from 9–11 and *then* have a look at my email." It makes it easier to be in control of things if you can give yourself rewards *later*; and if you get into a routine, as a parent probably suggested when you were an undergraduate (certainly mine did and I ignored them), then you'll find that your 'productivity' improves dramatically. My mother was right, dammit.

The Problem of No Breaks

Frequent distraction is *one* problem, but (in a sense) to *enough* distraction is another. As this piece suggests, it's also important that you give yourself downtime between Zoom/Teams/whatever sessions. I will try to remember to bake these into our 'lectures', but you should also *suggest* breaks if you feel yourself flagging! This also applies, however, to your wider degree: Masters degrees are *intense* and you need to give yourself permission for a timeout… stepping away from the computer and going for a walk, doing some knitting, whatever floats your boat!

How to *Read*

Although the guidance below, from Tim Squirrel's Guide for undergraduate essays, is about reading so that you can write an essay in history or philosophy, the advice works for *all* kinds of reading.

So, make sure that you cite the people who originally came up with the ideas you're presenting, and ideally think about how you would *differ from* or *improve upon* or *disagree with* them. The only way you can do this (and consequently, the only way to get a decent mark) is to do some reading.

1. **Look at the reading list**. If it's incredibly long, you probably won't want to (or be able) to read it all. However, that is *not* an excuse to not read any of it. Look through the list, identify if there are any readings marked as essential. Read them. If there aren't any essential readings, pick a few which look interesting and relevant, then read them.
2. Read some more. If the reading list is really short, you'll need to go beyond it. If it's long, this is still relevant. Look through the reference lists of the papers and books you've just read. See where their ideas came from. Mark out a few of the most promising-looking readings. Read them.
3. There is a difference between **reading to understand** the topic, and **reading to reference**. It is *totally fine* to use Wikipedia, lecture notes, etc to familiarise yourself with the key arguments and concepts. It is considerably less fine to cite them.
4. Books. **Do *not* read whole books**. It's a waste of your time. You won't remember any of it, it will drain all of your energy, and you only get one reference and viewpoint out of it. Read the intro and conclusion so that you get the gist of their argument. Pick a chapter from the contents page which looks like it's relevant to your essay. Read that. As above, find relevant references and follow them up.
5. Articles. **Read the abstract first**. Does it look like it's relevant? If not, don't waste your time. If it does, read it. Check the bibliography as above.

6. Read critically. **For the sake of all that is holy, read critically**. This is absolutely essential. Don't just stare at the pages and absorb them, bovine-like, for the purposes of regurgitation into your essay. **Think** about:

    1. The central claim the author is making. Usually there is only one, perhaps two. Summarise it in one sentence if you can.
    2. What is the frame of their argument? When in history is it set? Who are the key actors? Are they responding to another author? If so, what is the argument they're responding to? Try to position their argument in context. This allows you to:
    3. Critically assess the claims made. This obviously doesn't just mean 'say they're wrong'. They might well be wrong, but you'll need to find reasons for it. Generate a list of three reasons for each line of attack you want to take. Scrap the weakest two. If you think they're right, *why* are they right? Are there other authors who corroborate their claims? Are there logical reasons to prefer their argument?

Make sure you take notes on everything you read. Put page numbers in those notes. In fact, write down a few potentially useful (and ideally flexible) quotes verbatim.

Why am I Making This Hard for You?

Another challenge for many students is that they want the 'right' answer to how to do things. There's more academic literature on this, but for a thought-provoking look at why it might be a good idea for us to make your life *hard* then What IKEA and Our Education System Have in Common is an easy read.

Here's the summary:

1. IKEA is easy, but you don't learn anything.
2. IKEA is about getting things done/finishing.
3. IKEA is convenient, but it's not creative.
4. IKEA is standardised, but it is *actually* primed for hacking.

But there is a great response to the original post that adds nuance to this:

> The thing to keep in mind, though, is that following recipies is how we learn skills to start with. If you were teaching someone to cook, for example, you wouldn't throw the person into the kitchen and tell them to be creative with the food. Instead, you would teach them to follow recipies so that they can practice skills, and learn how to properly balance flavours and textures. Then, when they've mastered some recipies, you teach them how to mix up and re-combine recipies, and eventually come up with their own.

Thinking Like a Programmer

For the record, there are many things that *can't* be solved by code or coders, but there are many things that can be tackled by learning to think *like* a programmer. This can include:

1. **Understanding**: make sure that you actually understand the problem before you try to solve it. Try to explain it to someone else. Try to explain it to pet or stuffed animal. Write it down.
2. **Planning**: "Given X, what steps do I need to achieve Y?" You can start with comments, bullet points, whatever helps you to get the *skeleton* of an answer in place before you spend ages writing the first few lines of code *or* the first few lines of your essay.
3. **Dividing**: *never* try to solve a problem in one go. Break it down into little steps. Easy steps. Do the *easiest* one first (if you can separate it out from the *first* one). Check your solution works for that *part*. Take the next step. Check the two steps work together. Build from there. As programmers *and* Venture Capitalists would tell you: iterate! Same for an essay or written submission.
4. **Unsticking**: try to stay curious rather than getting angry or frustrated. Debugging is a step-by-step process: comment things out, add `print` statements, break it back down again into the basics and gradually re-add pieces until you can see what breaks.
5. **Practicing**: I like this quote "Practice. Practice. Practice. It'll only be a matter of time before you recognize that 'this problem could easily be solved with .' "

   "The art of debugging is figuring out what you really told your program to do rather than what you thought you told it to do."" — Andrew Singer

On a more practical level, here are common mistakes made by new programmers.

What This Means for You

- We will not *give* you the answer. This will be frustrating and annoying (particularly when you are already frustrated that something is not working) but by asking you questions we will try to teach you to solve problems *for yourself*.
- If you keep asking us the *same* questions you will get increasingly abrupt answers. You *should* be scared to ask us the *same* question for the 20th time. You should *not* be scared to admit that you're struggling with something.
- Sometimes there *is* no answer! Every year the data changes. The policies and issues change. I teach things a little differently. This is much more like the real world and sometimes I don't know what we'll find when we start coding.
- You need to work out how *you* learn best.

- You will need to think critically about what you are doing. There's a good article on How to write better essays which leads on to a blog on how to write better undergraduate essays (see above), You're obviously not undergraduates any more and we expect *more* of you, but as a starting point this is a good one; especially for those of you who are new to the UK way of teaching and learning.