1 Introduction

In this video, we talked about some functionalities of Overleaf. To get started with actually writing your Algo homework using Overleaf, we recommend reading the PDF together with the code. If you don't know how certain features are realized in code, double-click the PDF, and Overleaf will jump to the code corresponding to the feature. Sometimes the double-click feature does not work. If that happens, recompile the file and try again.

This PDF is meant to introduce you to some common features in LaTeX. The actual template that you can use for a problem set is in the file template.tex.

2 Sections and Subsections

Normally, an Algo problem set contains two to three problems. You can use the

\section{section name}

command to separate the problems. Everything you put in the section part will appear on the left in the file outline, so it's easy to navigate when the file is long. Feel free to copy the actual problems here, but it's not required.

We expect each question to have three subsections: the algorithm, runtime analysis, and proof of correctness. It's recommended to put them into three smaller subsections to make the structure clearer by using

\subsection{subsection name}

Here is a sample structure of one Algo problem:

2.1 Algorithm

Here, you can describe your algorithm for this particular problem. We encourage students to explain the algorithm in English instead of using pseudo-code. This way, if you make a simple mistake that ruins the correctness of your code (like an off-by-one error with a loop counter), we'll understand the intent of your solution. We only give points for what you write down, so be sure to specify your algorithm fully.

You can also use a separate line and make it bold to distinguish different sections like this:

Algorithm:

2.2 Runtime Analysis

We almost always ask for runtime analysis in homework problems. We require students to use LaTeX because it's a great tool for rendering math equations. This is a sample runtime analysis:

The preprocessing will take $O(n \log n)$ since we need to sort every object in increasing order. The loop will take $O(n^2)$ since each operation takes $O(n)$, and there are a total of n objects we are operating on. Therefore, the total runtime will be $O(n^2)$.

As a side note, this is way too general to get full points for runtime analysis. The runtime analysis should be much more problem specific.

2.3 Proof of Correctness

We encourage students to include graphs to help explain their algorithm or proof of correctness. Later in the semester, some algorithms might be easier to explain in graphs. Using both English and graphs helps to reduce the probability that we accidentally take points because of a typo in the explanation or completely misunderstand the algorithm. To insert a graph, we need to use the graphicx package, which we already imported.

Figure 1: Some random picture that took me $O(1)$ to draw.

For more details, check here.

3 Lists and Mathmode

1. We use the dollar sign to enter mathmode. If you need to write any equations, using mathmode will make it look more legible. The letters also look different in mathmode. For example, here's $O(n)$ (mathmode) and O(n) (not mathmode). Mathmode is helpful when inserting specific math symbols, which we'll show soon.

   - It's possible to have lists within lists. You can use

     \begin{itemize} and \begin{enumerate}

     based on your own choice [1]. This tutorial talks about further customizing the lists.

   - Here are a few math symbols that will be used pretty frequently. Again, double-click on the PDF to locate the code if needed.

     - $\leq, \geq, \in, \notin, \subseteq, \subsetneq, \wedge, \vee, \sim, \cup, \cap, \exists, \forall, \implies, \impliedby, \iff$
     - A_1(subscript), A^*(superscript), A_2^3(both).
       Note that a caret must be used to raise the * in mathmode, that's so we can also type things like $(f * g)(x)$ without raising the *.

[1] All \begin{X} should be closed by some \end{X}. Although most of the time, it's automatically handled by Overleaf.

  - $\sum_{i=0}^{n} a_i, \prod_{i=0}^{n} b_i$. These look fancier if you use double dollar signs, e.g.

$$\sum_{i=0}^{n} a_i.$$

  - $\frac{2a+3b^2}{\sqrt{c}}$
  - $a_1, a_2, \dots a_n, a \cdot b, A \setminus B$
  - $\mathbb{Z}$, $\mathbb{S}$, $\mathbb{R}$
  - ω, Ω (case sensitive)
  - $(a, b, c), [a, b, c], \{a, b, c\}$
  - $1 + (-1)^n = \begin{cases} 0, & \text{if } n \text{ odd} \\ 2, & \text{otherwise} \end{cases}$
  - Have parentheses around a fraction? Make sure to use \left and \right so you get $\left(\frac{a^2}{b^2+c^2}\right)^2$ instead of $(\frac{a^2}{b^2+c^2})^2$ (yuck).
  - Use \ to add some space like this: something   something

- I would recommend checking out these two websites when you're trying to figure out how to type a symbol. See this website for some common math symbols. See this website to find symbols by drawing them out (super helpful for me).

4 Efficiency is Everything

4.1 custom commands

Sometimes you may want to define your own commands. Here's the Overleaf tutorial. We'll talk about some simple examples here. There're two commands you can use:

\newcommand and \renewcommand

You must use the first one when you're defining a command that is not already defined and the second one if you want to overwrite existing commands. If you use the wrong one, errors will be reported in the logs.

Normally, we define all new commands after we import the packages or somewhere before the \begin{document}. This is an example of defining the absolute value (good to check the comments in the code here).

```
\newcommand{\abs}[1]{|#1|}
```

The 1 in the bracket means this command takes one input. Now, we can type in $|a|$ using this new command instead of manually typing $|a|$ (looks the same here but they're different in the code). It's helpful when the command is more complicated.

4.2 spacing

You may also want to skip the curly braces if you get tired of typing them. Generally, it's safe to do this if the thing inside the curly brace was only one character to begin with.

For instance, you can just say $\frac12$ instead of $\frac{1}{2}$ to get $\frac{1}{2}$. If you want $\frac{a}{b}$ though, you should use a space: $\frac ab$ (without the space, it will think fracab is one command). This also means that you should can get away with $nˆ2$ for n^2 (but don't get tempted into writing $nˆ10$ because that gives you $n^1 0$).

5 Math Equations

Now let's see a few ways to type in equations neatly. You can use \[\] if your equation can fit into one line (or it only has one line). For example:

$$S_{H_{ij}}(e_k) = e_k - \frac{2\langle e_i - e_j, e_k\rangle}{\langle e_i - e_j, e_i - e_j\rangle}(e_i - e_j)$$

If the equations you need to type in are long or have several steps, try using

\begin{align}

and using \\ to create a new line (works in general for texts too). For example:

$$\begin{align}
\omega_0(Av, Au) &= (Av)^T\Omega(Au) \tag{1}\\
&= (\lambda_1 v)^T\Omega(\lambda_2 u) \tag{2}\\
&= \lambda_1\lambda_2(v^T\Omega u) \tag{3}
\end{align}$$

By using the sign &, you can align different equations. If you don't want to number the equations, then you can use \begin{align*} instead like this:

$$\begin{align*}
\omega_0(Av, Au) &= (Av)^T\Omega(Au)\\
&= (\lambda_1 v)^T\Omega(\lambda_2 u)\\
&= \lambda_1\lambda_2(v^T\Omega u)
\end{align*}$$

Those equations are from some math homework. We promise that equations you need to type for Algo will be much simpler.

6 Other Things

If you really want to make your homework look fancy, here's one thing you can use.

Claim 1. *There exist* $u, v \in \mathbb{R} \setminus \mathbb{Q}$ *such that* $u^v \in \mathbb{Q}$.

Proof. Let $a = \sqrt{2}^{\sqrt{2}}$. If a is rational, then set $u = v = \sqrt{2}$ and we are done. Otherwise, consider $a^{\sqrt{2}} = \left(\sqrt{2}^{\sqrt{2}}\right)^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}\sqrt{2}} = \sqrt{2}^{2} = 2$. Then, set $u = a, v = \sqrt{2}$. $\square$

So now you can have some smaller sections in your homework. There're tons of templates and things you can do with LaTeX.

This video talks about some additional features that are probably good to know.

7 Some Final Words

If you ever get stuck on how to type something in, feel free to ask on Ed. It's also a good idea to search on Google. You can find some answers right away most of the time.