TEACHING LOW LEVEL CODING

Low level programming is all around us, and despite this, is widely forgotten about. It is the only way in which we interact with a computer, and it is the fastest way to run any program on a computer. Low level programming is a dying art. This is mainly due to high level languages and structures taking the leading role in education and software development. They are more tailored to how people think and so are much simpler to pick up. Despite this, low level programming is still incredibly relevant, now more than ever, and should not take a back seat. It is not until late education that we are taught about high level language being translated into something which can be understood by the computer being used. In a study undertaken by Smith and Webb (2000), they looked at how *meaningful learning* requires pre-existing knowledge to build upon. As we are not taught how the computer actually operates, when the topic of low-level programming emerges, the learner is forced to memorise information without proper understanding, and this will leave one’s mind very quickly. Their solution to this was to create a *glass box* interpreter based upon C, which would show a novice user exactly how the compilation process is undertaken. This is a strong parallel to my project, as I too wish to create this *glass box* model of an interpreter. The difference is that my *glass box* will be based on a custom programming language, as opposed to an existing language such as C. I think that I am likeminded to these researchers, and am out to solve the same problem. Their journal goes on to explain a great deal about the functionality of their *glass box* implementation and I am going to look into drawing parallels to these functionalities, this is because the results of the study when introduced to real students was very positive, and they are features which I believe will improve my current design greatly. I spoke about some of the features in prior, but these will add context and proof that this idea will work and will yield results.

Smith, P.A. and Webb, G.I. (2000) ‘The Efficacy of a Low-Level Program Visualization Tool for Teaching Programming Concepts to Novice C Programmers’, Journal of educational computing research, 22(2), pp. 187–215. doi:10.2190/N0VV-0P48-XJ9G-F8WV.