O Level Computer Studies 7010
Unit 5: Programming concepts

Recommended Prior Knowledge
Students need to have studied units 3 and 4 before beginning this unit.

Context
Students need to be able to distinguish between high-level and low-level programming languages and understand the concept of a program. Students should be able to describe algorithms using formal pseudocode structures.

Outline
Introducing the main requirements of a programming language, to allow manipulation of data of various types and structures, including control of input and output, and to provide for selection, repetition and subprogram intercommunication. Linking this with formally representing algorithms in pseudocode. A simple understanding of the functions of interpreters, compilers and assemblers, and an appreciation of the benefits offered by the existence of a range of languages, both high and low level.

AO	Learning outcomes	Suggested Teaching activities	Learning resources
3.2	Program structures including – sequence, selection and repetition. Pseudocode structures.	Introduce the idea of a program and the different routes that can be taken through program code using the suggested pseudocode structures of Repeat ... Until. If ... Then ... Else … Endif. Case of ... ... Otherwise ... Endcase. While ... Do ... Endwhile. For … to … next It is suggested that students work through past examples (from earlier exam papers) to see the level required. This should be linked into 3.1 (unit 4) since algorithmic design and pseudocode structures should be taught side by side.	http://www.unf.edu/~broggio/cop2221/2221pseu.htm introduction to pseudocode some simple examples http://perl.about.com/od/beginningperl/a/072604.htm is good explanation of pseudocode for teachers L+W 9.1 and 9.6
	High level and low level languages and their uses.	Introduction to different types of programming languages and their uses. Discuss the features of high level languages (portable, close to English language, easier to modify and write, etc.) and low level languages (e.g.	http://www.bbc.co.uk/scotland/education/bitesize/standard/computing/systems/software_rev5.shtml introduction to

AO	Learning outcomes	Suggested Teaching activities	Learning resources
		machine code and assembly code). Also discuss suitability of high and low level languages for certain applications due to the features (e.g. machine code used to write games, operating systems, etc.)	programming languages http://www.klbschool.org.uk/ict/gcse/theory/software/language.htm different levels of programming language http://www.webopedia.com/TERM/H/high_level_language.html different types and levels of programming language http://inventors.about.com/library/weekly/aa072198.htm more about FORTRAN extension work L+W 9.7
	Translation programs including compilers, interpreters and assemblers.	Introduce the three types of translators and highlight the differences between them.	http://www.bbc.co.uk/scotland/education/bitesize/standard/computing/systems/software_rev6.shtml http://www.computingstudents.com/notes/computer_systems/assemblers_and_compilers.php more about assemblers and compilers http://www.play-hookey.com/computers/language_levels.html a more in depth article about translators L+W 9.7
	User and technical documentation	Introduce the different types of documentation, that required by people who are going to use a system and that required by those responsible for improving and maintaining a solution in working order or for developing the solution to meet new needs. This links to the coursework project. The different features of user documentation (e.g. how to load the	http://www.theteacher99.btinternet.co.uk/theteacher/gcse/newgcse/others/documentation.htm introduction to the different types of documentation L+W 22.1 and 22.2

AO	Learning outcomes	Suggested Teaching activities	Learning resources
		software, input requirements, etc.) and technical documentation (e.g coding, flow charts, hardware requirements, test runs, etc.) should be looked at in depth. It is suggested that a real example is taken and the students taken through the documentation features – this could be linked into any of the earlier sections as appropriate.	