Software Design & Development – Higher Checklist

Topic	Software Design and Development	Tick
Development methodologies	Describe and compare the development methodologies: • iterative development process • agile methodologies	
Analysis	Identify the: • purpose • scope • boundaries • functional requirements of a problem that relates to the design and implementation at this level, in terms of: • inputs • processes • outputs	
Design	Identify the data types and structures required for a problem that relates to the implementation at this level.	
	Read and understand designs of solutions to problems at this level, using the following design techniques: • structure diagrams • pseudocode	
	Exemplify & implement efficient design solutions, using a recognised design technique, showing: • top level design • the data flow • refinements	
	Describe, exemplify and implement user-interface design, in terms of input and output, using a wireframe.	
Implementation (data types and structures)	Describe, exemplify and implement appropriately the following structures: • parallel 1D arrays • records • arrays of records	
Implementation (computational constructs)	Describe, exemplify & implement appropriate constructs in a procedural high-level (textual) language: • parameter passing (formal and actual) • the scope of local and global variables	
	sub-programs/routines, defined by their name and arguments (inputs and outputs): • functions & procedures	
	pre-defined functions (with parameters): • to create substrings *(string[start:end])* • to convert from character to ASCII *(ord())* and vice versa *(chr())* • to convert floating-point numbers to integers *(int())* • modulus *(%)*	
	file handling: • sequential CSV (open, create, read, write, close) • txt files (open, create, read, write, close)	
	Read and explain code that makes use of the above constructs	
Implementation (algorithm specification)	Describe, exemplify and implement standard algorithms using 1D arrays or arrays of records: • count occurrences • find minimum and maximum • linear search	

Testing	Describe, exemplify & implement a comprehensive test plan to show functional requirements are met.
	• Identify syntax, execution, and logic errors at this level.
	• Describe and exemplify debugging techniques:
	  ○ dry runs
	  ○ trace tables/tools
	  ○ Breakpoints
	  ○ Watchpoints	

Evaluation	Describe, identify and exemplify the evaluation of a solution in terms of: • fitness for purpose • efficient use of coding constructs • usability • maintainability • robustness	