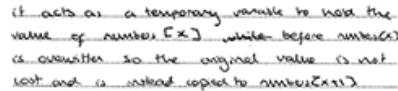


Mark scheme

Question			Answer/Indicative content	Marks	Guidance
1			<p>1 mark per bullet to max 3 e.g.</p> <ul style="list-style-type: none"> Initialising a new board with 100 squares Placing players on square 1 at the start Setting a new 30-minute timer Rolling two dice and displaying the total Moving the counters the number of places on the dice Choosing an instruction card and displaying the message Checking if a player has landed on square 100 Checking if the timer is 0 	3	<p>Allow other suitable examples that are relevant to the scenario.</p> <p><u>Examiner's Comments</u></p> <p>Some candidates struggled to apply decomposition to identify parts of the scenario that could be broken down into sub problems. Examples needed to relate to the gameplay for the given scenario and needed to be qualified in some way. For example, 'Dice' on its own was insufficient, whereas 'Generating dice score' was sufficient.</p>
			Total	3	
2			<p>1 mark each to max 2 e.g.</p> <ul style="list-style-type: none"> Connect to database Access usernames in file/database Check username against file/database Hash password Access password/hash in file/database Check password entered/hashed vs stored Output result 	2	<p>Allow other suitable subprocedures that link to the scenario.</p> <p><u>Examiner's Comments</u></p> <p>This question was also generally well answered, but answers had to be given in the context of the scenario. Thinking procedurally is specification point 2.1.3 which requires suitable procedures for a given context to be identified. The context was explicitly for logging in to a system, not for creating user accounts or setting up/validating password construction. The most common responses were checking if the username existed and checking whether the username/password combination was correct.</p>
			Total	2	
3		i	<p>1 mark per bullet</p> <ul style="list-style-type: none"> by reference will reorder the contents of the array 	3	<p><u>Examiner's Comments</u></p> <p>Many candidates struggled to go beyond recall of definitions for calling by reference and calling by value and</p>

			<ul style="list-style-type: none"> ...so the new order can be accessed by the main program / so will be saved when the procedure ends by value will change the array only in this procedure ... and so would need to return the array. 		<p>struggled to apply it to the code given and to provide a detailed explanation.</p> <p>The bubble sort was defined as a procedure and not a function, so if numbers had been passed by value, a copy of the array would have been passed, and any changes made would not have been kept after the procedure had completed execution.</p>
		ii	A loop that repeats a fixed / set number of times	1	<p><u>Examiner's Comments</u></p> <p>Most candidates could accurately define a 'count controlled loop' as one that repeated a predefined number of times, although some candidates gave an ambiguous response that was equally applicable to a conditional loop.</p>
		iii	<ul style="list-style-type: none"> To temporarily hold a value (for numbers[x])... ...while it is being transferred from one position to another...in the array numbers To stop values over writing each other 	3	<p><u>Examiner's Comments</u></p> <p>Many candidates found it difficult to explain the purpose of the <code>holdValue</code> variable in context. Where candidates achieved some of the marks, they most frequently identified <code>holdValue</code> as a temporary store that was required to prevent accidental overwriting of data during the swap process. Relatively few were able to accurately describe how the variable allowed the contents of <code>dataArray[x]</code> and <code>dataArray[x+1]</code> to be swapped.</p> <p>Exemplar 1</p>  <p>This exemplar very clearly states exactly how and why the variable <code>holdValue</code> is required.</p>
		iv	<ul style="list-style-type: none"> Add a (second outer) loop That will repeat for each pass / repeat until the flag is set to true at the end of a pass 	2	<p><u>Examiner's Comments</u></p> <p>Many candidates found it challenging to apply knowledge of a bubble sort to the code given. While a pleasing number identified the need to have an</p>

					outer loop, there were far fewer who were able to expand on this to explain that this was required to repeat the process for the required number of passes, or until no swaps had occurred during a pass.
			Total	9	
4			<p>1 mark for each component e.g.</p> <ul style="list-style-type: none"> • Allocating cards to each player • Generating the deck • Managing whose turn it is to play • Checking won 	3	<p>Accept any reasonable component</p> <p><u>Examiner's Comments</u></p> <p>The specification requires candidates to be able to identify elements of computational thinking. As such, candidates are expected to be able to think procedurally and to be able to identify the components of a problem. While analysis of the problem given in context led most candidates to identify valid components, many struggled to read the scenario and to give relevant points. For instance, many reiterated aspects of checking if a move was valid, which was already given in the question.</p>
			Total	3	