Task	Expected response		Additional guidance	Marks available	
2	**Software design and development**				
2a	**1 mark** each for design including: ♦ assignment: random number between 1 and 10 ♦ selection: if random number = 1, set customer's bill to 0 ♦ selection: if random number >= 2 and <=6, set customers bill to 50%		Random number assignment must include range. There is no requirement for the design to show the bill does not change for random numbers between 7 and 10	3	Design (3)
2b	Fixed loop for number of items entered by user			1	Implementation (15)
	Input Validation	conditional loop used		1	
		correct loop condition	Until item = "c" OR item = "t" OR item = "b" or While until item ≠ "c" AND item ≠ "t" AND item ≠ "b"	2	
		input of item type	Award 1 mark if not implemented within input validation loop	1	
		error message		1	
	Running total for bill using correct prices for items			1	
	Random number: ♦ store random integer (1-10) ♦ display random number			2	
	If random number = 1 set bill total to 0		Accept <2 as condition	1	
	If random number >= 2 and <= 6 set bill to 50% of bill		1 for condition 1 for correct calculation	2	
	Display final bill to 2 decimal places			1	
	If statements match design:			2	

Task	Expected response	Additional guidance	Marks available	
2	**Software design and development**			
	♦ nested or else if statements used to add up bill items ♦ separate if statements used to determine final bill			

Task	Expected response	Additional guidance	Marks available	
2	**Software design and development**			
2ci	Table completed correctly with all three possible outputs: ♦ 0 ♦ 4.5 ♦ 9	For reference total = 9: (c = 2.25 + t = 1.85 + t = 1.85 + b = 3.05)	1	Testing (3)
	Printed evidence of test run producing one of the above outputs.	Both inputs and outputs should be printed	1	
cii	Table completed with one exceptional value.		1	
2d	Evaluation of the following Efficiency **(2 marks)** ♦ two comments on the efficient use of coding constructs Robustness **(1 mark)** ♦ how robust the program is, including if it copes with unexpected inputs Readability **(1 mark)** ♦ the readability of the candidate's own code	Candidate may describe either efficiencies or inefficiencies in their code Evaluation must contain an element of evaluation rather than simple statements of terms. For example "I have used white space to highlight structures in my program" not "I have used white space". The candidate's code must also show evidence of this for a mark to be awarded.	4	Evaluation (4)