SYSC 4101

Laboratory 4

Function `findPrice(itemType, itemCode, quantity, weight)` is a function used in a grocery store software system. It takes four inputs: an `itemType`, an `itemCode`, a `quantity` and a `weight`.

The item type (parameter `itemType`) indicates the type of item being purchased while the code (parameter `itemCode`) uniquely identifies an item of a given type with seven digits. In other words, the pair (`itemType`, `itemCode`) uniquely identifies an item, while two different items of two different types can have the same code. The quantity purchased is provided by input `quantity`. The weight of the item purchased is provided by input `weight`. Parameter `itemType` specifies how the other three parameters need to be used or ignored (see below).

All parameters are integers:

- `itemType` is a one-digit code.
- `itemCode` is a seven-digit code.
- `quantity` is not greater than 50.
- `weight` is not greater than 50 (kg).

Function `findPrice`() accesses a database to find and display the unit price, a short description of the item, and the total price of the item corresponding to pair (`itemType`, `itemCode`) for the purchased quantity/weight. The function displays an error message (and returns) if any of the four input values is incorrect or if their combination is incorrect (see below).

`itemType`	Interpretation
0	Ordinary grocery items such as bread, magazine, and soup. - The `quantity` matters (a `quantity` value strictly positive is expected, otherwise it is an error). - The `weight` is not required (a `weight` value of zero is expected, and any other value results in an error).
1	Variable-weight items such as meats, fruits, and vegetables. - The `weight` matters (a `weight` value strictly positive is expected, otherwise it is an error). - The `quantity` is not required (a `quantity` value of zero is expected, and any other value results in an error).
2	Health related items such as Tylenol. - The `quantity` matters (a `quantity` value strictly positive is expected, otherwise it is an error). - The `weight` is not required (a `weight` value of zero is expected, and any other value results in an error).
3	Discount coupon. - The `itemCode` indicates the coupon code. - The amount is fetched from the database. - The `weight` and `quantity` are not required (values of zero for `weight` and `quantity` are expected, and any other value for these parameters results in an error).
other	Ignored: corresponds to incorrect input values for `itemType`.

For example:

- Input (0, 1234567, 3, 0) refers to the purchase of a quantity of 3 of ordinary grocery (uniquely identified by number 1234567).
    - But Input (0, 1234567, 3, 12) results in an error.
- Input (1, 1234567, 0, 12) refers to the purchase of 12 weight units of a variable-weight item (uniquely identified by number 1234567).
    - Notice that two items can have the same `itemCode` if they belong to different categories.
- Input (2, 1234567, 3, 0) refers to the purchase of a quantity of 3 of a health-related item (uniquely identified by number 1234567).
- Input (3, 1234567, 0, 0) refers to a discount coupon with code 1234567.

Exercise

Apply the **Input Domain Modeling** testing method and identify parameters, possibly environment variables, characteristics and blocks (including base blocks).
When reporting on the use of the testing method:

- Make sure to clearly identify the parameters, the possible environment variables, the characteristics, the blocks, the base blocks, the constraints as discussed in class.
- Make sure to use the template illustrated in course slides for structuring this information whereby the test model provides information with indentation.
- Make sure any assumption you are making is clearly stated.

Given your test model (characteristics, blocks, …), create test frames:

- To satisfy the Each-block criterion
- To satisfy the Pair-wise criterion
- To satisfy the Base-block criterion

Rubric

	Excellent (5)	Competent (3)	Needs work (1)
Input Domain Modeling 30%	No major parameter (and environment variable), characteristic, block, constraint is missing (or is incorrect) in the solution. The solution is valid.	Some parameters, characteristics, blocks, or constraints are missing, or elements of the solution are invalid.	Many parameters, environment variables, characteristics, blocks, or constraints are missing (or are incorrect), or many elements of the solution are invalid.
Use of criteria 40%	The criteria are applied without any error and with proper observations.	Some errors are made when applying the criteria.	The principles of the criteria are not followed, resulting in many errors.
Clarity of solution 15%	The solution, including the decisions being made, as reported by parameters, characteristics, blocks, constraints, as well as any assumptions made is clear.	The solution, including decisions being made, is in general clear, though several points are vague.	The wording of the solution (e.g., parameters, characteristics, blocks, constraints) is vague, preventing a clear understanding of the solution.
Reporting template 15%	The reporting template is fully and correctly used.	Several aspects of the reporting template are not mastered.	The reporting template is not applied.