TYPESCRIPT LAB

Overview: This lab includes three parts to give you lots of TypeScript practice. Complete the entire lab in one project using one `.ts` file.

TALLEST MOUNTAIN

- Declare an interface called `Mountain` that contains the following properties:
  - `name` - string
  - `height` - number
- Declare an array called `mountains` which is an array of type `Mountain`.
- Fill the array with the following mountains:

name	height
Kilimanjaro	19341
Everest	29029
Denali	20310

- Declare a function called `findNameOfTallestMountain`. It takes one parameter, an array of `Mountain` objects. It returns a string, the name of the tallest mountain in the given array. If the array argument is empty, return an empty string ("").
- Call `findNameOfTallestMountain`, passing it your `mountains` array as an argument.
- Store the result of the function call (the return value) in a variable and then console.log the variable. (Hint: It will print “Everest”.)

PRODUCTS

- Declare an interface called `Product` that contains the following properties:
  - `name` - string
  - `price` - number
- Declare an array called `products` which is an array of type `Product`.
- Fill the array with a few products of your own choosing.
- Declare a function called `calcAverageProductPrice`. It takes one parameter, an array of `Product` objects. It returns a number, the average price of all the products provided as an argument. If the array argument is empty, return 0.
- Call `calcAverageProductPrice`, passing it your `products` array as an argument.
- Store the result of the function call (the return value) in a variable and then console.log the variable.

continued on next page...

INVENTORY

- Declare an interface called `InventoryItem` that contains the following properties:
    - `product` - `Product` (from above)
    - `quantity` - number
- Declare an array called `inventory` which is an array of type `InventoryItem`.
- Fill the array with the following information.

product.name	product.price	quantity
motor	10.00	10
sensor	12.50	4
LED	1.00	20

- Declare a function called `calcInventoryValue`. It takes one parameter, an array of `InventoryItem` objects. It returns a number, the total value of all the products in the inventory array provided as an argument. If the array argument is empty, return 0.
    - This is calculated as follows: For each InventoryItem take the product price times the quantity. Add these together for all the items. For the above data, the total will be 10.00 x 10 + 12.5 x 4 + 1.00 * 20 = 170.
- Call `calcInventoryValue`, passing it your `products` array as an argument.
- Store the result of the function call (the return value) in a variable and then console.log the variable. (Hint: It prints 170).

Tests

1. `Mountain` interface exists with name (string) and height (number) properties.
2. `mountains` array exists with given data.
3. `Product` interface exists with name (string) and price (number) properties.
4. `products` array exists with several objects of data.
5. `InventoryItem` interface exists with product (Product) and quantity (number) properties.
6. `inventory` array exists with given data.
7. `findNameOfTallestMountain` takes Mountain array parameter and returns correct string.
8. `calcAverageProductPrice` takes Product array parameter and returns correct number.
9. `calcInventoryValue` takes InventoryItem array parameter and returns correct number.
10. All of the functions (findNameOfTallestMountain, calcAverageProductPrice, and/or calcInventoryValue) that have been created are called correctly and the result stored and logged.