**Hardware Store**

Implement functionality for hardware stores.

*Notes:*

* There is no need to specify an error message when **throwing** an Error.
* **Use single quotes for strings: 'string'!!!!!!!!!?!?!!**

**Class descriptions**

**class Product**

* Properties
  + id - should be generated automatically and be different for different products????
  + manufacturer - string with length between 1 and 20 symbols (inclusive)
  + model - string with length between 1 and 20 symbols (inclusive)
  + price - positive, non-zero number
  + **Throw** if any of the properties are invalid
* Methods:
  + getLabel() - returns a string: what should be written on the label
    - read on :)

**class SmartPhone extends Product**

* Properties:
  + screenSize - positive, non-zero number
  + operatingSystem - string with length between 1 and 10 symbols (inclusive)
  + **Throw** if any of the properties are invalid
* Methods:
  + getLabel() - returns a string: what should be written on the label
    - SmartPhone - MANUFACTURER MODEL - \*\*PRICE\*\*

**class Charger extends Product**

* Properties:
  + outputVoltage - number between 5 and 20 (inclusive)
  + 5 - number between 100 and 3000 (inclusive)
    - its in milliamperes if you are asking
  + **Throw** if any of the properties are invalid
* Methods:
  + getLabel() - returns a string: what should be written on the label
    - Charger - MANUFACTURER MODEL - \*\*PRICE\*\*

**class Router extends Product**

* Properties:
  + wifiRange - positive, non-zero number
  + lanPorts - positive, non-zero **integer** number
  + **Throw** if any of the properties are invalid
* Methods:
  + getLabel() - returns a string: what should be written on the label
    - Router - MANUFACTURER MODEL - \*\*PRICE\*\*

**class Headphones extends Product**

* Properties:
  + quality - string, should be either high, mid or low
    - **Throw** if quality is invalid
  + hasMicrophone - boolean
    - convert true-like javascript values to true and false-like to false???
* Methods:
  + getLabel() - returns a string: what should be written on the label
    - Headphones - MANUFACTURER MODEL - \*\*PRICE\*\*

**class HardwareStore**

* Properties:
  + name - string with length between 1 and 20 symbols (inclusive)
    - **Throw** if invalid
  + products - array of unique products currently in storage
    - should be empty when the store is created
* Methods:
  + stock(product, quantity) - adds new products
    - product should be a valid Product instance
    - quantity should be a positive, non-zero integer number
    - **Throw otherwise**
    - **Should provide chaining**
  + sell(productId, quantity) - sells products
    - quantity should be a positive, non-zero integer number
    - there should be at least quantity products with id productId available in the store
    - **Throw otherwise**
    - **Should provide chaining**
  + getSold() - returns the amount of money earned from selling in the current store
  + search(pattern) - returns an array of unique products containing pattern in their model or manufacturer name
    - perform **case insensitive** search
    - each element in the array should have 2 keys:
      * product - the product instance
      * quantity - the available quantity of that product
  + search(options) - advanced search, same as above
    - options is an object with **optional** keys:
      * manufacturerPattern - string, should be contained in manufacterures **(case sensitive)**
      * modelPattern - string, should be contained in models **(case sensitive)**
      * type - string - SmartPhone, Charger, Router or Headphones - the product should be of the specified type
      * minPrice - number - the product should not be cheaper than minPrice
      * maxPrice - number - the product should not be more expensive than maxPrice

**Sample usage**

const result = solve();

const phone = result.getSmartPhone('HTC', 'One', 903, 5, 'Android');

console.log(phone.getLabel()); // SmartPhone - HTC One - \*\*903\*\*

const headphones = result.getHeadphones('Sennheiser', 'PXC 550 Wireless', 340, 'high', false);

const store = result.getHardwareStore('Magazin');

store.stock(phone, 1)

.stock(headphones, 15);

console.log(store.search('senn'));

/\*

[ { product:

Headphones { ... },

quantity: 15 } ]

\*/

console.log(store.search({type: 'SmartPhone', maxPrice: 1000});

/\*

[ { product:

SmartPhone { ... },

quantity: 1 } ]

\*/

console.log(store.search({type: 'SmartPhone', maxPrice: 900});

/\* [] \*/

store.sell(headphones.id, 2);

console.log(store.getSold()); // 680

**Solution template**

function solve() {

// Your classes

return {

getSmartPhone(manufacturer, model, price, screenSize, operatingSystem) {

// returns SmarhPhone instance

},

getCharger(manufacturer, model, price, outputVoltage, outputCurrent) {

// returns Charger instance

},

getRouter(manufacturer, model, price, wifiRange, lanPorts) {

// returns Router instance

},

getHeadphones(manufacturer, model, price, quality, hasMicrophone) {

// returns Headphones instance

},

getHardwareStore(name) {

// returns HardwareStore instance

}

};

}

// Submit the code above this line in bgcoder.com

module.exports = solve; // DO NOT SUBMIT THIS LINE