## Computer

You need to implement the class hierarchy for a computer business, here are the classes you should create and support:

* Keyboard class that contains:
  + manufacturer - string property for the name of the manufacturer
  + responseTime - number property for the response time of the Keyboard
* Monitor class that contains:
  + manufacturer - string property for the name of the manufacturer
  + width - number property for the width of the screen
  + height - number property for the height of the screen
* Battery class that contains:
  + manufacturer - string property for the name of the manufacturer
  + expectedLife - number property for the expected years of life of the battery
* Computer - **abstract** class that contains:
  + manufacturer - string property for the name of the manufacturer
  + processorSpeed - a number property containing the speed of the processor in GHz
  + ram - a number property containing the RAM of the computer in Gigabytes
  + hardDiskSpace - a number property containing the hard disk space in Terabytes
* Laptop - class **extending** the Computer class that contains:
  + weight - a number property containing the weight of the Laptop in Kilograms
  + color - a string property containing the color of the Laptop
  + battery - an instance of the Battery class containing the laptop's battery. There should be a getter and a setter for the property and validation that the passed in argument is actually an instance of the Battery class.
* Desktop - concrete class **extending** the Computer class that contains:
  + keyboard - an instance of the Keyboard class containing the Desktop PC's Keyboard. There should be a getter and a setter for the property and validation that the passed in argument is actually an instance of the Keyboard class.
  + monitor - an instance of the Monitor class containing the Desktop PC's Monitor. There should be a getter and a setter for the property and validation that the passed in argument is an instance of the Monitor class.

Attempting to instantiate an abstract class should throw an Error, attempting to pass an object that is not of the expected instance (ex. an object that is not an instance of Battery to the laptop as a battery) should throw a TypeError.

### Example

|  |
| --- |
| computer.js |
| **function** *createComputerHierarchy*() {  *//****TODO: implement all the classes, with their properties* return** {  ***Battery***,  ***Keyboard***,  ***Monitor***,  ***Computer***,  ***Laptop***,  ***Desktop*** } } |

You are asked to submit **ONLY the function** that returns an object containing the above-mentioned classes.

### Bonus:

In order to achieve a better code reuse, it's a good idea to have a base abstract class containing common information - check the classes, what common characteristics do they share that can be grouped in a common base class.