## 5.Assembly Line

Create a function that **returns** a **library of decorator functions**. They can be used to **compose** different functionality in a **car object** that they receive as an argument.

Your solution must **return an object**, containing **three decorator functions**:

hasClima – compose air conditioning controls into the passed-in object. This function takes an **object as a parameter** and adds to it the following properties:

* temp – **number** with default value **21**;
* tempSettings – **number** with default value **21**;
* adjustTemp – **function** which takes **no arguments**. If temp is less than tempSettings, this function adds 1 to temp. If temp is more than tempSettings, it decreases temp by 1. If temp and tempSettings are equal, the function does nothing.

hasAudio – compose audio player functionality into the passed-in object. This function takes an **object as a parameter** and adds to it the following properties:

* currentTrack – **object** with properties name (string) and artist (string). The default value is null;
* nowPlaying – **function**, which **prints** on the console the text:

`Now playing '${currentTrack.name}' by ${currentTrack.artist}` , where name and artist are properties of the currentTrack object. If currentTrack is null, this function does nothing.

hasParktronic – compose parking aid functionality into the passed in object. This function takes an **object as a parameter** and adds to it the following properties:

* checkDistance – **function**, which takes a **single argument** distance (number) and **prints** a message on the console, depending on its value:

distance < 0.1 – "Beep! Beep! Beep!"

0.1 <= distance < 0.25 – "Beep! Beep!"

0.25 <= distance < 0.5 – "Beep!"

In any other case, print an **empty string**.

### Input

Your **solution** will receive **no arguments**. All the methods in the returned library must take an **object as an argument**. Any methods that you compose into this object must meet the input requirements listed in the description above.

### Output

Your **solution** must **return an object** containing the **three decorators** described above.

### Example

|  |  |
| --- | --- |
| **Setup** | |
| const assemblyLine = createAssemblyLine();  const myCar = {  make: 'Toyota',  model: 'Avensis'  }; | |
| **Input** | **Output** |
| assemblyLine.hasClima(myCar);  console.log(myCar.temp);  myCar.tempSettings = 18;  myCar.adjustTemp();  console.log(myCar.temp); | 21  20 |
| **Input** | **Output** |
| assemblyLine.hasAudio(myCar);  myCar.currentTrack = {  name: 'Never Gonna Give You Up',  artist: 'Rick Astley'  };  myCar.nowPlaying(); | Now playing 'Never Gonna Give You Up' by Rick Astley |
| **Input** | **Output** |
| assemblyLine.hasParktronic(myCar);  myCar.checkDistance(0.4);  myCar.checkDistance(0.2); | Beep!  Beep! Beep! |
| **Input** | **Output** |
| console.log(myCar); | {  make: 'Toyota',  model: 'Avensis',  temp: 20,  tempSettings: 18,  adjustTemp: [Function],  currentTrack: {  name: 'Never Gonna Give You Up',  artist: 'Rick Astley'  },  nowPlaying: [Function],  checkDistance: [Function]  } |