1- Define **Person Object** Contains the following Properties: ID:1, Name : ”Empty” ,

Define **employee object** which contains Salary property and make it’s **prototype Person Object**

Define HREmployee object which contains location Property and make it’s **Prototype employee object**

- Test prototype chain in console for HREmployee and Employee objects
- Try to access person ID and Person Name using HREmployee object
- Define Name And ID Properties with values For HREmployee Object then test if it accessible with person object
- Define Age Property with Person Object and test if it accessible with HREmployee Object

2- A- Define Shape Base Abstract Class which contains color property and define PrintColor method and CalcArea and Calcperimeter which will return Zero in Shape Base Class , Define them on Shape prototype object

B- Define Rect Class Which inherits from Shape Abstract Class (prototypal inheritance)

Define Width and Height Properties for Rect Class

C- Define Square Class Which inherits from Rect Class

- override CalcArea , calcperimeter , printColor , toString which will display color , area and perimeter in rect and square classes

- create array object which will contains set of objects from rect and square classes then display it’s areas

3- Define static property and static method like following case for Rect and Square classes to get number of objects created from rect and square Types

```
function Person(_id,_name){
     this.ID = _id;
     this.Name = _name;
     Person.NumberOfObjects++;
 }

Person.NumberOfObjects = 0; // Static or Class Property
Person.DisplayNumberOFObjects = function () { // Static or Class Method
return this.NumberOfObjects; };
```