Private variables using closure

```
var add = (function() {
        var count=0;
        return function() {return count+=1;};
})();


console.log(add());
console.log(add());
console.log(add());
```

```
1
2
3
[Finished in 0.1s]
```

In the above section of code, ‘add’ is a variable whose value is allocated to a self-invoking function. But we can see that it is returning a function call to child function, meaning it can be treated as function instead of a variable. In this example ‘count’ is behaving like private variable, inside the ‘add’ function. When add() is invoked the ‘count’ is set to 0 and returns child function which has logic to change the value of ‘count’ variable, but ‘count’ has its scope as parent block, which is fine as inner blocks have access to outer block’s scope. This is called Javascript Closure. Now in simple terms ‘count’ variable can only be changed by function called by return of add(), therefore making it private variable inside of a function.