```
//O QUASE ENCAPSULAMENTO O JAVASCRIPT:
//Quando se fala de Encapsulamento, o Javascript não é nenhuma grande maravilha, pois, por ser uma linguagem fracamente tipada e
ainda por cima ser muito dinâmica, é muito fácil alterar os tipos e dados dos objetos em Javascript.
//Porém existem algumas funções built-in que podem nos ajudar a lidar com os nossos objetos evitando mudanças bruscas nas suas chaves
e até nos seus dados:

//FUNÇÃO PREVENTEXTENEXTENSIONS() - PREVENINDO EXTENSÕES NO OBJETO:
//A função preventExtensios() como o próprio nome diz prevê que as chaves de um objeto possam ser extendidas, podemos modificar os
valores das chaves normalmente e podemos até deletar algumas chaves já existentes, mas nunca poderemos adicionar novas chaves.
Podemos usá-la da seguinte forma...
const produto = Object.preventExtensions( //Note aqui que usamos a função no momento da inicialização do objeto, mas podemos usar
depois do objeto estar inicializado também...
    {
        nome: 'Qualquer',
        preco: 1.99,
        tag: 'promocao'
    } //O objeto não poderá receber chaves novas além dessas
    )

produto.nome = 'Borracha' //Veja que podemos mudar os valores...
produto.descricao = 'Essa é uma borracha que serve para apagar' //Mas não podemos adicionar novas chaves...
delete produto.tag //Mas podemos deletar chaves já existentes...
console.log('1)', produto)



//FUNÇÃO ISEXTENSIBLE() - VERIFICA SE UM OBJETO É EXTENSÍVEL OU NÃO:
//A função isExtensible() verifica se um objeto criado pode ser extendido - ou seja, pode receber chaves novas - ou não...
console.log('\n2)', Object.isExtensible(produto)) //Usamos ela sempre após o Object, assim como a preventExtensions(), ela vai
retornar "true" se o objeto for extensível ou "false" se o objeto não for extensível...



//FUNÇÃO SEAL() - SELANDO AS CHAVES DE UM OBJETO:
//A função seal() vai além da preventExtensions(), além de impedir que chaves novas possam ser criadas, a seal(), impede que as
chaves possam ser deletadas também, porém os valores das chaves ainda poderão ser modificados normalmente...
const pessoa = {nome: 'Gabriel', idade: 30}
```

```
Object.seal(pessoa) //Aqui nós selamos o objeto "pessoa", suas chaves não poderão ser adicionadas e nem removidas...

pessoa.sobrenome = 'Ferreira' //Veja que uma chave nova não pôde ser adicionada...
delete pessoa.nome //A chave já existente não pôde ser removida...
pessoa.idade = 31 //O valor pôde ser alterado normalmente...
console.log('\n3)', pessoa)



//FUNÇAO ISSEALED() - VERIFICANDO SE UM OBJETO ESTÁ SELADO OU NÃO:
//A função isSealed() verifica se um objeto foi selado ou não, se tiver sido selado ela retorna "true", se não tiver sido, ela
retorna "false"...
console.log('\n4)', Object.isSealed(pessoa))



//FUNÇÃO FREEZE() - CONGELANDO TOTALMENTE UM OBJETO:
//A função freeze() congela totalmente um objeto, depois de criado nada poderá ser feito com ele, suas chaves não poderão ser
adicionadas nem removidas, e até mesmo seus valores não poderão ser trocados...
const copo = Object.freeze(
    {
        volume: 'cheio',
        temperatura: 'quente'
    }
)

copo.volume = 'metade' //O valor não pôde ser modificado...
delete copo.temperatura //A chave não pôde ser deletada...
copo.canudo = true //Nem uma nova chave pôde ser adicionada...

console.log('\n5)', copo)



//FUNÇÃO ISFROZEN() - VERIFICANDO SE UM OBJETO ESTÁ CONGELADO:
//A função isFrozen() verifica de um objeto foi congelado pela função "freeze()", retornando "true" se o objeto tiver sido congelado,
ou "false" se ele não tiver sido...
```

```
console.log('\n6)', Object.isFrozen(copo))
```

RESULTADO NO CONSOLE...

```
[Running] node "c:\Users\Almoxarifado\Documents\javascript\arquivos_das_aulas\093-O_Quase_Encapsulamento_do_Javascript.js"
1) { nome: 'Borracha', preco: 1.99 }

2) false

3) { nome: 'Gabriel', idade: 31 }

4) true

5) { volume: 'cheio', temperatura: 'quente' }

6) true

[Done] exited with code=0 in 0.114 seconds
```