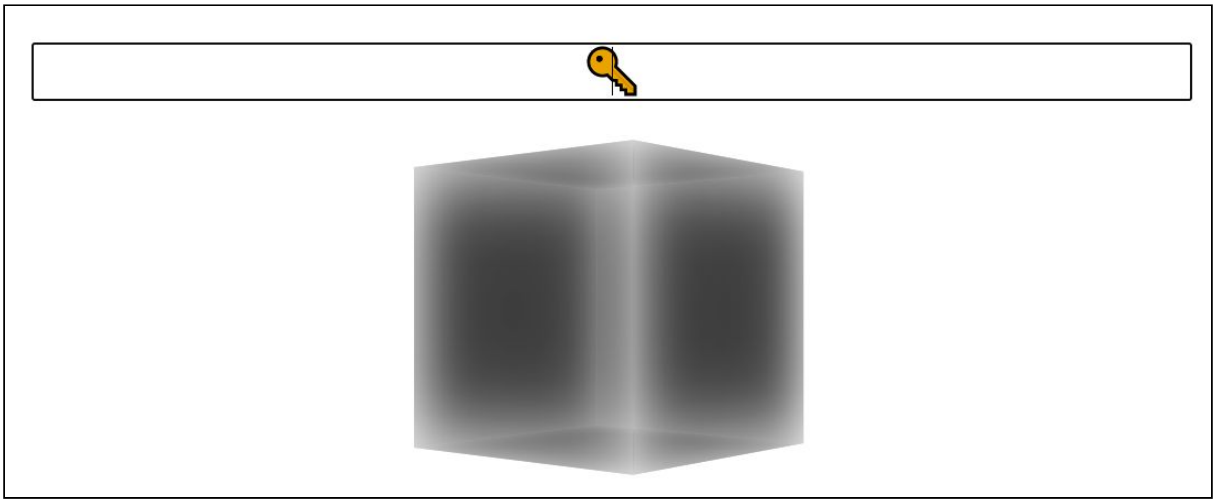
Desafio 3

Desafio: Encontramos um cofre em html, você consegue descobrir a senha que gera a mensagem "Acesso Garantido"?

Resolução do Desafio:

1) Foi fornecido um código em HTML o qual possui um campo para entrada de senha.



2) Para analisar o código fonte do arquivo abri o código em outra página.
Após analisar o código foram encontradas algumas informações importantes:

2.1 - No código há um comentario dizendo basicamente como javascript é complicado de entender. Isso dá a entender então que os códigos em javascript contidos no código fonte devem ser analisados porque provavelmente devem conter alguma informação relevante.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Super Mega Hyper Blaster Ultra seguro Cofre em JavaScript</title>
<!--
Quem disse que uma implementação de segurança em client-side é ruim? kkk
Só de ter que lidar com javascript qualquer pessoa já desistiria, logo, somente quem tem a senha pode acessar as informações.
É impossível alguém olhar para esse código e tentar entendê-lo, nem eu sei o que ele faz mais, só sei que ele funciona.
Além disso, é cientificamente comprovado que olhar para javascript durante mais de 10 minutos faz a pessoa perder um neurônio
a cada 20 minutos. Então qualquer hacker espertinho vai ficar completamente burro depois de tentar arrombar esse cofre
-->
```

2.2 - Analisando ainda o código, há uma parte dele que demonstra o que será mostrado na tela quando a senha for colocada no campo solicitado. Então, saberemos assim que a senha inserida é a correta.

```
}
body.granted>#result::before {
  content: "Acesso Garantido\AMeu segredo é: odeio javascript";
  white-space: pre;
  color: green;
}
```

2.3 - O outro ponto importante no código seria procurar por códigos em javascript. Existem 3 funções em Javascript no código:

2.3.1 - Essa primeira função aparentemente recebe a senha e depois retorna ***true*** ou ***false.***

```
function x(x) {
  function lol(n,u){return r=n===u,r}function kkk(n,u){return n+u}function aa(n,u,r){return n+r+u}function bb(){return"QuN"}function cc(){return"0YTR"}function dd(){return"c2Nya"}function ee()
{return"ZyBq"}function ff(){return"YXZh"}function gg(){return"ZnVj"}function hh(){return"Gawj0"}function ii(){return"aJnmj0"}function jj(){return"QuN"}function kk(){return"C90LMg0Y"}function ll()
{return"Gne0"}function mm(){return"Nani"}function nn(){return"a2lu"}function oo(){return"Jasd"}function pp(){return"Igu2"}function qq(){return"XB0"}function rr(){return"Juaw"}function ss()
{return"Ns=="}function tt(){return"QstCw0YHRhtGA0L"}function uu(){return"Htha"}function ww(){return"1337"}function vv(){return"1234"}function xx(){return"Ju12"}function yy(){return"28js"}function zz()
{return"Unw0"}for(a=0;1e3!=a;a++);for(a=0;1e3!=a;a++);xx=kkk(kkk(gg(),nn()),aa(ee(),dd(),ff()),qq());
    return lol(xx, x);
}
```

2.3.2 - Essas duas outras funções recebem o resultado ***true*** ou ***false*** e retorna a mensagem que informa se a senha inserida está correta ou não.

```
function open_safe() {
  keyhole.disabled = true;
  password = keyhole.value;
  if (!(x(password))) return document.body.className = 'denied';
  document.body.className = 'granted';
}
function save() {
  plaintext = Array.from(content.value).map(c => c.charCodeAt());
  localStorage.content = JSON.stringify(plaintext.map((c,i) => c ^ password[i % password.length]));
}
```

3) Para analisar melhor a função ***function x(x)*** coloquei-a em um Desofuscador para que a função aparecesse de uma maneira ordenada e linear para melhor entendimento.

Obs: O Desofuscador usado foi: https://lelinhtinh.github.io/de4js/

```
function x(x) {
    function lol(n, u) {
        return r = n === u, r
    }

    function kkk(n, u) {
        return n + u
    }

    function aa(n, u, r) {
        return n + r + u
    }

    function bb() {
        return "QuN"
    }

    function cc() {
        return "0YTR"
    }

    function dd() {
        return "c2Nya"
    }

    function ee() {
        return "ZyBq"
    }
```

```
function ff() {
    return "YXZh"
}

function gg() {
    return "ZnVj"
}

function hh() {
    return "Gawj0"
}

function ii() {
    return "aJnmj0"
}

function jj() {
    return "QuN"
}

function kk() {
    return "C90LMg0Y"
}

function ll() {
    return "Gne0"
}
```

```
function mm() {
    return "Nani"
}

function nn() {
    return "a2lu"
}

function oo() {
    return "Jasd"
}

function pp() {
    return "Igu2"
}

function qq() {
    return "XB0"
}

function rr() {
    return "Juaw"
}

function ss() {
    return "Ns=="
}
```

```
function tt() {
    return "QstCw0YHRhtGA0L"
}

function uu() {
    return "Htha"
}

function ww() {
    return "1337"
}

function vv() {
    return "1234"
}

function xx() {
    return "Ju12"
}

function yy() {
    return "28js"
}

function zz() {
    return "Unw0"
}
```

```
    for (a = 0; 1e3 != a; a++);
    for (a = 0; 1e3 != a; a++);
    xx = kkk(kkk(gg(), nn()), aa(ee(), dd(), ff()), qq());
    return lol(xx, x);
}
```

4) Ao analisar a função apresentada no **tópico 3** pode se perceber que a função recebe o valor da senha, e dentro do próprio código, pelas chamadas de funções a senha correta também é gerada, e então o código faz uma comparação entre a senha gerada pelo próprio código e a senha fornecida pelo usuário, e então, se a função retornar ***true*** quer dizer que as duas senhas comparadas são iguais, se retornar ***false*** quer dizer que as senhas comparadas são diferentes.
   Abaixo o código comentado com as explicações das funções:

```
function x(x) { // Função que recebe a senha fornecida pelo usuário
    function lol(n, u) { // Faz a comparação da senha fornecida pelo usuário e a senha gerada pela função
        return r = n === u, r
    }

    function kkk(n, u) { // Soma os dois valores recebidos como parâmetro
        return n + u
    }

    function aa(n, u, r) { // Soma os três valores recebidos como parâmetro
        return n + r + u
    }

    function bb() {
        return "QuN"
    }

    function cc() {
        return "0YTR"
    }

    function dd() {
```

```
    return "c2Nya"
}

function ee() {
    return "ZyBq"
}

function ff() {
    return "YXZh"
}

function gg() {
    return "ZnVj"
}

function hh() {
    return "Gawj0"
}

function ii() {
    return "aJnmj0"
}

function jj() {
    return "QuN"
}

function kk() {
    return "C90LMg0Y"
}

function ll() {
    return "Gne0"
}

function mm() {
    return "Nani"
}
```

```
function nn() {
    return "a2lu"
}

function oo() {
    return "Jasd"
}

function pp() {
    return "Igu2"
}

function qq() {
    return "XB0"
}

function rr() {
    return "Juaw"
}

function ss() {
    return "Ns=="
}

function tt() {
    return "QstCw0YHRhtGA0L"
}

function uu() {
    return "Htha"
}

function ww() {
    return "1337"
}

function vv() {
```

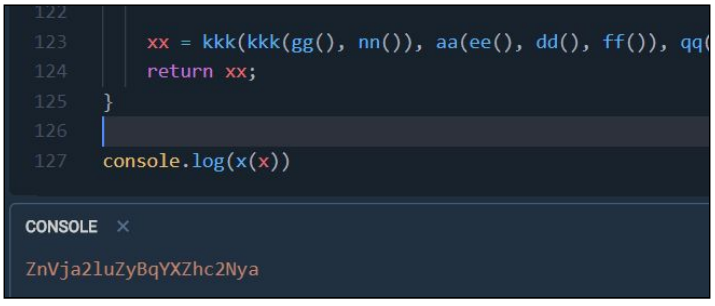
```
        return "1234"
    }

    function xx() {
        return "Ju12"
    }

    function yy() {
        return "28js"
    }

    function zz() {
        return "Unw0"
    }
    for (a = 0; 1e3 != a; a++); //Não está sendo utilizado no
código
    for (a = 0; 1e3 != a; a++); //Não está sendo utilizado no
código
    /* São funções dentro de funções. Basicamente ocorrerá o
seguinte:
    1) kkk(kkk(ZnVj,a2lu), aa(ZyBq,c2Nya, YXZh),XB0)
    2) kkk(kkk(ZnVja2lu), aa(ZyBqYXZhc2Nya),XB0)
    3) kkk(ZnVja2luZyBqYXZhc2Nya)
    */
    xx = kkk(kkk(gg(), nn()), aa(ee(), dd(), ff()), qq()); //
Guarda o valor retornado em uma variável
    return lol(xx, x);
```

5) Depois de entender o código no final do código em vez de retornar a comparação entre a senha fornecida pelo usuário e a senha gerada pela função, retornei somente a senha gerada pela função.



6) Por fim, ao colocar a senha fornecida aparece os seguintes dizeres:

7) Ademais, pode-se observar que essa senha está em base64. Utilizando um decodificador a frase seguinte é mostrada:

fucking javascr