```
//TRY.. CATCH.. THROW.. e FINALLY:
//O try, catch, throw e finally, literalmente traduzido por tentar, pegar, jogar e finalmente, são palavaras reservadas em muitas lin
guagens de programação para o tratamento de erro, onde:
// try: é usado para colocar o código que submiteremos ao possível erro...
// catch: para pegar o erro, quando o erro acontece a função catch é chamada e será executado o que estiver dentro de catch...
// throw: o throw vai trazer uma resposta ao erro, ele serve como um return, mas somente para tratamento de erros...
// finally: é muito opcional o finally, ele é um bloco de código que vai executar independente do erro ter acontecido ou não...
//Vejamos alguns exemplos...

//USANDO TRY CATCH THROW E FINALLY APRESENTANDO UMA MENSAGEM DE ERRO COMUM...
function tratamento1(erro) { //Perceba que criamos o throw antes, afinal ele será chamado para apresentar um resultado diferente...
    throw 'Houve um erro de sintaxe no seu código, dê uma olhada melhor no seu objeto e refatore...'; //Perceba que podemos escrever
diretamente após o throw o que desejamos mostrar para o usuário como resposta, poderíamos colocar qualquer execução de código aqui, e
m execuções grandes teríamos que colocar um bloco de código...
}

function Gritar(obj) { //Essa função coloca qualquer string em upper case...
    try { //Perceba eu o try envolve a execução da String...
        console.log(obj.name.toUpperCase() + '!!!'); // O erro é na chamada da chave name, que foi passada como "nome"...
    } catch (erro) { //Caso o erro aconteça...
        tratamento1(erro);
    } finally { //Sempre será mostrado independente de o erro acontece ou não...
        console.log('Fim da depuração');
    }
}

const obj = {nome: 'Roberto'}; //Perceba que a função espera por uma chave "name" e não "nome"...
Gritar(obj);



//USANDO TRY CATCH THROW E FINALLY APRESENTANDO UMA MENSAGEM DE ERRO DETALHADA PELA CLASSE ERROR DO JAVASCRIPT...
function tratamento2(erro) {
    throw new Error('O ocorreu um erro na chamada do objeto, troque a chave nome por "name"...'); //Perceba que podemos gerar um erro
 customizado á partir do construtor Error, ele vai imitar a saída de um erro como é mostrado pelo próprio Javascript, porém, podemos
customizá-lo passando a mensagem que desejarmos...
```

```
}

function imprimirNomeGritando(obj) {
    try {
        console.log(obj.name.toUpperCase() + '!!!');
    } catch (erro) {
        tratamento2(erro);
    } //Neste caso não usamos o finally...
}

imprimirNomeGritando(obj);



//TRATANDO ERRO E MOSTRANDO ONDE A FALHA ACONTECEU...:
function erro(num) { //Perceba que criamos um throw onde o programa mostra que o erro aconteceu mas ainda consegue arrumar o problema
 e mostrar na tela o resultado correto, porém, por ter acontecido um erro o programa finaliza...
    throw console.log('Aconteceu um erro! Você passou uma string quando deveria ter passado um Number, seu resultado é:\n', num = Num
ber(num) + Number(num), '\nDa próxima vez passe um Number!!!\nFim do Programa');
}

function somarPorEleMesmo(num) {
    try { //Perceba que temos um try que compara se o valor passado é uma string ou um number, se não for number ele gera um error...
        comparador = Number(num);
        if (num === compardor) {
        return console.log('A soma do número por ele mesmo é', num + num);
        } else {
            return Error();
        }
    } catch {
        erro(num);
    }
}

somarPorEleMesmo('5');
```

RESULTADO DO TRATAMENTO DE ERRO 1...

```
[Running] node "c:\Users\User\Documents\JAVASCRIPT\ARQUIVOS_DAS_AULAS\051-Try_Catch_Throw.js"
Fim da depuração

c:\Users\User\Documents\JAVASCRIPT\ARQUIVOS_DAS_AULAS\051-Try_Catch_Throw.js:11
    throw 'Houve um erro de sintaxe no seu código, dê uma olhada melhor no seu objeto e refatore...'; //Perceba que podemos escrever
diretamente após o throw o que desejamos mostrar para o usuário como resposta, poderíamos colocar qualquer execução de código aqui, e
m execuções grandes teríamos que colocar um bloco de código...
    ^
Houve um erro de sintaxe no seu código, dê uma olhada melhor no seu objeto e refatore...
(Use `node --trace-uncaught ...` to show where the exception was thrown)

[Done] exited with code=1 in 0.164 seconds
```

RESULTADO DO TRATAMENTO DE ERRO 2...

```
[Running] node "c:\Users\User\Documents\JAVASCRIPT\ARQUIVOS_DAS_AULAS\051-Try_Catch_Throw.js"
c:\Users\User\Documents\JAVASCRIPT\ARQUIVOS_DAS_AULAS\051-Try_Catch_Throw.js:31
    throw new Error('O ocorreu um erro na chamada do objeto, troque a chave nome por "name"...'); //Perceba que podemos gerar um erro
 customizado á partir do construtor Error, ele vai imitar a saída de um erro como é mostrado pelo próprio Javascript, porém, podemos
customizá-lo passando a mensagem que desejarmos...
    ^

Error: O ocorreu um erro na chamada do objeto, troque a chave nome por "name"...
    at tratamento2 (c:\Users\User\Documents\JAVASCRIPT\ARQUIVOS_DAS_AULAS\051-Try_Catch_Throw.js:31:11)
    at imprimirNomeGritando (c:\Users\User\Documents\JAVASCRIPT\ARQUIVOS_DAS_AULAS\051-Try_Catch_Throw.js:38:9)
    at Object.<anonymous> (c:\Users\User\Documents\JAVASCRIPT\ARQUIVOS_DAS_AULAS\051-Try_Catch_Throw.js:42:1)
    at Module._compile (internal/modules/cjs/loader.js:1072:14)
    at Object.Module._extensions..js (internal/modules/cjs/loader.js:1101:10)
    at Module.load (internal/modules/cjs/loader.js:937:32)
    at Function.Module._load (internal/modules/cjs/loader.js:778:12)
    at Function.executeUserEntryPoint [as runMain] (internal/modules/run_main.js:76:12)
    at internal/main/run_main_module.js:17:47

[Done] exited with code=1 in 0.145 seconds
```

RESULTADO DO TRATAMENTO DE ERRO COM ALTERAÇÃO CAUSADA PELO THROW...

```
[Running] node "c:\Users\User\Documents\JAVASCRIPT\ARQUIVOS_DAS_AULAS\051-Try_Catch_Throw.js"
Aconteceu um erro! Você passou uma string quando deveria ter passado um Number, seu resultado é:
 10
Da próxima vez passe um Number!!!
Fim do Programa

c:\Users\User\Documents\JAVASCRIPT\ARQUIVOS_DAS_AULAS\051-Try_Catch_Throw.js:48
    throw console.log('Aconteceu um erro! Você passou uma string quando deveria ter passado um Number, seu resultado é:\n', num = Num
ber(num) + Number(num), '\nDa próxima vez passe um Number!!!\nFim do Programa');
    ^
undefined
(Use `node --trace-uncaught ...` to show where the exception was thrown)

[Done] exited with code=1 in 0.16 seconds
```