<https://www.codewars.com/kata/predict-your-age/train/javascript>

KATA SOLUCIÓN 1 PARA EDADES:

function predictAge(age1,age2,age3,age4,age5,age6,age7,age8){

let ages = [age1,age2,age3,age4,age5,age6,age7,age8];

let sum = 0;

ages.forEach(function(value) {

let currentValue = value\*value;

sum = sum + currentValue;

});

sum = Math.sqrt(sum);

sum = Math.trunc(sum /2);

return sum;

}

SOLUCIÓN 2:

function predictAge(age1,age2,age3,age4,age5,age6,age7,age8){

let ages = [age1,age2,age3,age4,age5,age6,age7,age8];

return Math.trunc(Math.sqrt(ages.map(value => value\*value).reduce((a,b)=>a+b))/2);

}

-----------------------------------------------------

1. **GIT TAGS:**

Etiquetar puntos específicos de la historia como importantes. (Ejemplo: versiones)

Listar: git tag

Patrón particular: git tag -l 'v1.4.2.\*'

Existen etiquetas ligeras y anotadas

Ligera: es un puntero a una confirmación específica sin más información

Anotada: tienen comprobación (nombre anotador, correo, fecha, mensaje y pueden ser firmadas) git tag -a v1.4 -m 'my version 1.4' ahora, firmada git tag -s v1.5 -m 'my signed 1.5 tag'

Verificar etiqueta firmada: git tag -v v1.4.2.1

Etiquetar un commit que ya se hizo: git tag -a v1.2 -m 'version 1.2' 9fceb02

Las etiquetas se deben subir con git push:

git push origin v1.5 o subir todas git push origin --tags

1. **GIT REBASE VS GIT MERGE:**



* El rebase unifica las ramas dejando un árbol lineal de commits.
* El merge requiere un commit más para unificar las ramas, mientras que el rebase unifica de una vez.
* El rebase unifica las ramas perdiendo el historial de los commits de la otra rama . Esto puede resultar bien importante cuando se necesite saber el historial de commit.

1. **GIT FETCH:**

Git fetch trae todos los cambios de las ramas remotas pero no las une a la rama actual, es decir no es un pull.

1. **GIT SHORTCUTS:**

Alias para acortar texto en git, ejemplo.

git config --global alias.ci commit

Ahora en vez de usar git commit, usas git ci