What’s the way that we Refactor our

if(){}

and

else if(){}?

1 - We create a brand new else if(){}. (Or it’s only the if(){} and else if(){}, A brand new if(){})

2 – We put the conditions of the else if(){}s that have nearly the same code stored in them, in the brace of the new else if(){} with the||logical operator (Or if you’re clever enough for it, Write a condition that includes both of them).

3 – We put the shared code of the if(){} and else if(){} in the curled brace|{}|of the new else if(){}.

4 – If there’s any piece of code that in the curled brace|{}|of the if(){} and else if(){} that’s different, Then

4.1 : We use the ? operator

4.2 : We put the condition of either (one) of if(){} or else if(){} to the left of the ? operator, And we put the code that would’ve been executed in their condition to its right. (With a colon|:|at its end, Naturally)

4.3 : We put the code that would’ve been executed in the other scenario to the right of the colon|:|.

Note : We ONLY we do this Refactoring with either a if(){}, Or two else if(){}.

Note 2 : We don’t we this Refactoring if the conditions or the codes if the if(){} or else if(){} is too different from each other. (One having more conditions, More code than the other, etc….)

Note 3 : Don’t worry about the code that you put to the right of the colon|:|of the ? operator happening in any circumstances other than the ones you specified, The conditions that we set in the new else if(){} ensure that.

Note 4 : We don’t Refactor if(){} and else if(){} (or two else if(){}s) with each other, IF there are if(){}, else if(){} and else{} inside of them, Instead we Refactor the inner if(){}, else if(){} and else{} that are inside them.

Note 5 : Let me give an example of this.

I have a variable that has Math.random() \* 21 stored in it.

The job of the user is to guess the number.

If it’s correct, They get a congratulation.

If the guess is too high they get a massage saying that and lose a point.

If the guess is too low they get a massage saying that and lose a point.

The code in its normal state looks like this (I’ve excluding the non-important parts of the code) :

var user\_guess = Number(document.querySelector(`.number\_form`).value)

var score = 20

var the\_number = Math.Random() \* 21

if(user\_guess === number) {

console.log(`congragulations`)

}

else if(user\_guess > number) {

console.log(`Too high`)

score--

}

else if(user\_guess < number) {

console.log(`Too small`)

scor--

}

As you can see, The code that we stored in the else if(){} is too similar, With the only difference being the text in console.log(), Thus we Refactor it into:

var user\_guess = Number(document.querySelector(`.number\_form`).value)

var score = 20

var the\_number = Math.Random() \* 21

if(user\_guess === number) {

console.log(`congragulations`)

}

// Refactored here

else if(user\_guess > number|| user\_guess < number) {

user\_guess > number ? console.log(`Too high`):console.log(`Too small`);

score--

}

The code that we put to the right of the colon of |:|of the ? operator will ONLY activate when

user\_guess < number

Because of the condition we set in the else if(){}.