Math.random() function returns a floating-point, pseudo-random number in the range 0 to less than 1 (inclusive of 0, but not 1) with approximately uniform distribution over that range — which you can then scale to your desired range. The implementation selects the initial seed to the random number generation algorithm; it cannot be chosen or reset by the user.

Math.ceil() function always rounds a number up to the next largest integer.

Math.floor() function returns the largest integer less than or equal to a given number.

```
function getRandomInt(min, max) {

  min = Math.ceil(min);

  max = Math.floor(max);

  return Math.floor(Math.random() * (max - min) + min);
//The maximum is exclusive and the minimum is inclusive
```

concatenate - powiązać

ellipsis - wielokropek

Selecting from Many Options with Switch Statements

If you have many options to choose from, use a *switch* statement. A `switch` statement tests a value and can have many *case* statements which define various possible values. Statements are executed from the first matched `case` value until a `break` is encountered.

Here is an example of a `switch` statement:

```
switch(lowercaseLetter) {
  case "a":
    console.log("A");
    break;
  case "b":
    console.log("B");
    break;
}
```

Basic JavaScript: Escape Sequences in StringsPassed

Quotes are not the only characters that can be *escaped* inside a string. There are two reasons to use escaping characters:

1. To allow you to use characters you may not otherwise be able to type out, such as a carriage return.
2. To allow you to represent multiple quotes in a string without JavaScript misinterpreting what you mean.

We learned this in the previous challenge.

Code	Output
`\'`	single quote
`\"`	double quote
`\\`	backslash
`\n`	newline
`\r`	carriage return
`\t`	tab
`\b`	word boundary
`\f`	form feed

Using Objects for Lookups

Objects can be thought of as a key/value storage, like a dictionary. If you have tabular data, you can use an object to "lookup" values rather than a `switch` statement or an `if/else` chain. This is most useful when you know that your input data is limited to a certain range.

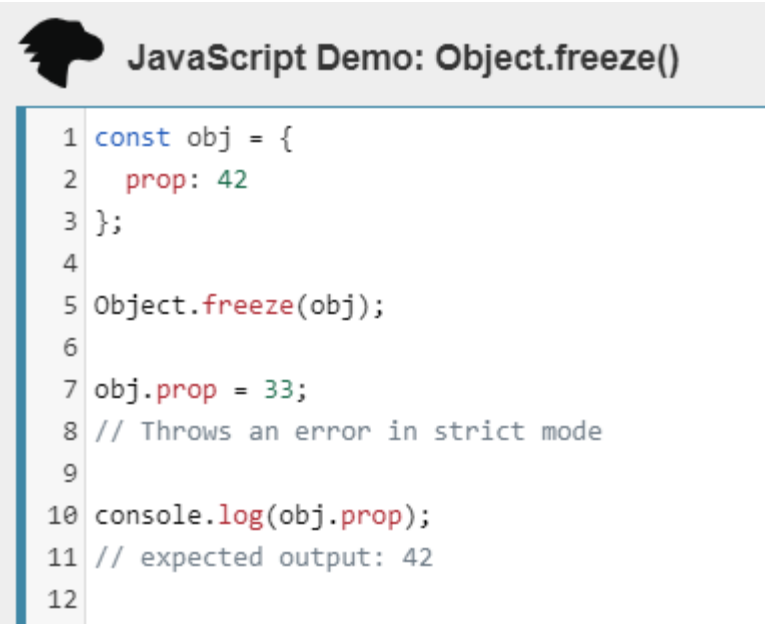
Here is an example of a simple reverse alphabet lookup:

```
var alpha = {
  1:"Z",
  2:"Y",
  3:"X",
  4:"W",
  ...
  24:"C",
  25:"B",
  26:"A"
};
alpha[2]; // "Y"
alpha[24]; // "C"

var value = 2;
alpha[value]; // "Y"
```

mainstay - ostoja, filar

The **`Object.freeze()`** method **freezes** an object. A frozen object can no longer be changed; freezing an object prevents new properties from being added to it, existing properties from being removed, prevents changing the enumerability, configurability, or writability of existing properties, and prevents the values of existing properties from being changed. In addition, freezing an object also prevents its prototype from being changed. `freeze()` returns the same object that was passed in.



JavaScript Array reduce() Method

Example

Subtract the numbers in the array, starting from the beginning:

```
var numbers = [175, 50, 25];

document.getElementById("demo").innerHTML = numbers.reduce(myFunc);

function myFunc(total, num) {
  return total - num;
}
```

```
var testArray = [3, 2, 5, 15];
testArray.reduce(function(total, current,) {
  return total + current
});
25
```

JavaScript String charAt() Method

Example

Return the first character of a string:

```
var str = "HELLO WORLD";
var res = str.charAt(0);
```

Definition and Usage

The charAt() method returns the character at the specified index in a string.

The index of the first character is 0, the second character is 1, and so on.

Tip: The index of the last character in a string is *string*.length-1, the second last character is *string*.length-2, and so on (See "More Examples").