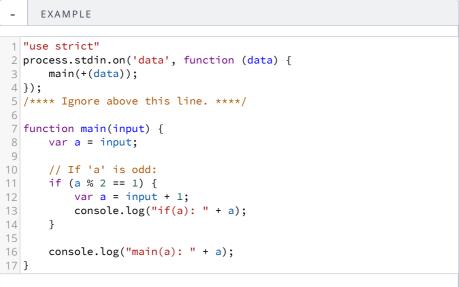
Day 1: Let and Const

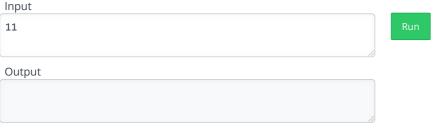


Variable Declaration Keywords

var

We use the *var* keyword to declare variables. The scope of a variable declared using this keyword is within the context wherever it was declared. For variables declared outside any function, this means they are globally available throughout the program. For variables declared within a function, this means they are only available within the function itself.





Solution

Click Run above to execute the given code. It works in the following way:

- 1. Variable a is declared in the *main* function using the *var* keyword and initialized with the given value, 11.
- 2. a % 1 evaluates to *true* because a = 11 is odd, so we enter the *if* block.
- 3. Variable a is declared a second time inside the if block (still using the var keyword) and initialized with a value of 11+1=12. We print the value of a=12.
- 4. We exit the *if* block and print the value of a in *main*. This value is 12 because the scope of the initial declaration of a in *main* includes the *if* block.

Table Of Contents var let

const



let

We use the *let* keyword to declare variables that are limited in scope to the block, statement, or expression in which they are used. This is unlike the *var* keyword, which defines a variable globally or locally to an entire function regardless of block scope.

```
EXAMPLE
 1 "use strict"
2 process.stdin.on('data', function (data) {
       main(+(data));
4 });
5 /**** Ignore above this line. ****/
6
7
   function main(input) {
8
       let a = input;
9
       // If 'a' is odd:
10
        if (a % 2 == 1) {
11
            // Increment 'a' by 1
12
            let a = input + 1;
13
14
            console.log("if(a): " + a);
15
       }
16
17
       console.log("main(a): " + a);
18 }
  Input
  11
  Output
 Solution
 Click Run above to execute the given code. It works in the following way:
 1. Variable a is declared in the main function using the let keyword and initialized
   with the given value, 11.
 2. a \% 1 evaluates to true because a = 11 is odd, so we enter the if block.
 3. Variable a is declared a second time inside the if block (again using the let
   keyword) and initialized with a value of 11 + 1 = 12. We print the value of
   a = 12.
 4. We exit the if block and print the value of a in main. Because we used the let
   keyword for both declarations and the scope of the second declaration of a was
   limited to the if block, the value of a in main is still 11.
 It's important to note that you cannot redeclare a variable declared using the let
 keyword within the same scope as the original variable. An attempt to do this raises
 an Error, as demonstrated by the code below.
 1 "use strict"
```

5 /**** Ignore above this line. ****/

main(+(data));

3
4 });

2 process.stdin.on('data', function (data) {

```
function main(input) {
    let a = input;

// This will throw "SyntaxError: Identifier 'a' has already been
let a = input + 1;

console.log(a);

Input

Output

Output
Output
```

const

We use the *const* keyword to create a *read-only* reference to a value, meaning the value referenced by this variable cannot be reassigned. Because the value referenced by a constant variable cannot be reassigned, JavaScript *requires* that constant variables always be initialized.

```
EXAMPLE
 Click Run below to see what happens when you declare a constant variable without
 initializing it.
 1 "use strict"
 2 process.stdin.on('data', function (data) {
       main(+(data));
4 });
 5 /**** Ignore above this line. ****/
 7 function main(input) {
 8
       const a = input;
9
10
       // This will throw "SyntaxError: Missing initializer in const de
11
       const b;
12
       console.log(a);
13
14 }
  Input
  11
  Output
```