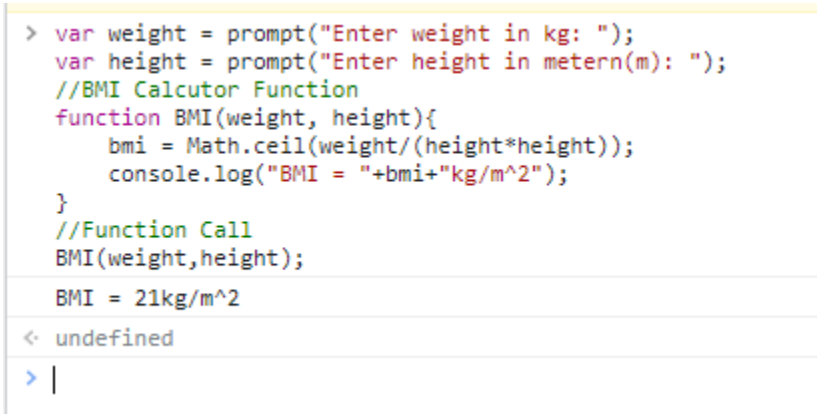
BMI Calculator

$$BMI = \frac{weight\,(kg)}{height^2\,(m^2)}$$

```
var weight = prompt("Enter weight in kg: ");
var height = prompt("Enter height in metern(m): ");
//BMI Calcutor Function
function BMI(weight, height){
    bmi = Math.ceil(weight/(height*height));
    console.log("BMI = "+bmi+"kg/m^2");
}
//Function Call
BMI(weight,height);
```

BMI = 21kg/m^2

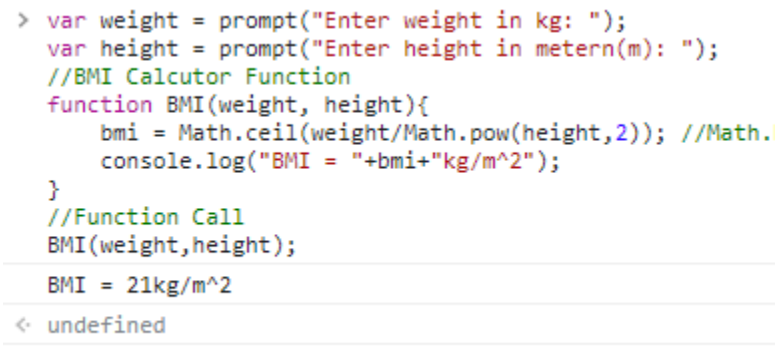


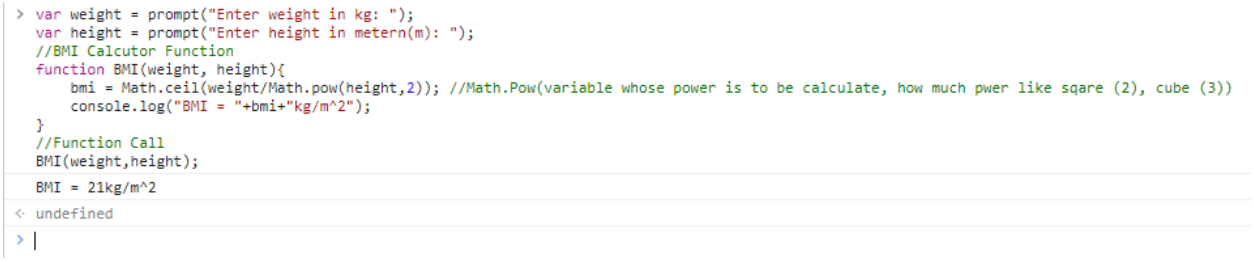
Method 2:

Use Power Function

```
var weight = prompt("Enter weight in kg: ");

var height = prompt("Enter height in metern(m): ");

//BMI Calcutor Function

function BMI(weight, height){

   bmi = Math.ceil(weight/Math.pow(height,2)); //Math.Pow(variable whose power is to be calculate, how much pwer like sqare (2), cube (3))

   console.log("BMI = "+bmi+"kg/m^2");

}

//Function Call

BMI(weight,height);
```

BMI = 21kg/m^2





https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math/pow

Math.pow()

The **`Math.pow()`** static method, given two arguments, *base* and *exponent*, returns $base^{exponent}$.

```
console.log(Math.pow(7, 3));
// expected output: 343

console.log(Math.pow(4, 0.5));
// expected output: 2

console.log(Math.pow(7, -2));
// expected output: 0.02040816326530612
//                  (1/49)

console.log(Math.pow(-7, 0.5));
// expected output: NaN
```

Syntax

```
Math.pow(base, exponent)
```

Parameters

`base`

The base number.

`exponent`

The exponent used to raise the `base`.

Return value

A number representing the given base taken to the power of the given exponent.

Description

The **`Math.pow()`** function returns the `base` to the `exponent` power, as in $base^{exponent}$, the `base` and the `exponent` are in decimal numeral system.

Because `pow()` is a static method of `Math`, use it as `Math.pow()`, rather than as a method of a `Math` object you created. (`Math` has no constructor.) If the base is negative and the exponent is not an integer, the result is NaN.

Examples

Using Math.pow()

```
// simple
Math.pow(7, 2);    // 49
Math.pow(7, 3);    // 343
Math.pow(2, 10);   // 1024

// fractional exponents
Math.pow(4, 0.5);  // 2 (square root of 4)
Math.pow(8, 1/3);  // 2 (cube root of 8)
Math.pow(2, 0.5);  // 1.4142135623730951 (square root of 2)
Math.pow(2, 1/3);  // 1.2599210498948732 (cube root of 2)

// signed exponents
Math.pow(7, -2);   // 0.02040816326530612 (1/49)
Math.pow(8, -1/3); // 0.5

// signed bases
Math.pow(-7, 2);   // 49 (squares are positive)
Math.pow(-7, 3);   // -343 (cubes can be negative)
Math.pow(-7, 0.5); // NaN (negative numbers don't have a real square root)

// due to "even" and "odd" roots laying close to each other,
// and limits in the floating number precision,
// negative bases with fractional exponents always return NaN
Math.pow(-7, 1/3); // NaN
```