Hoisting

JavaScript **Hoisting** refers to the process whereby the interpreter appears to move the *declaration* of functions, variables, classes, or imports to the top of their scope, prior to execution of the code.

Any of the following behaviors may be regarded as hoisting:

1. Being able to use a variable's value in its scope before the line it is declared. ("Value hoisting")
2. Being able to reference a variable in its scope before the line it is declared, without throwing a `ReferenceError`, but the value is always `undefined`. ("Declaration hoisting")
3. The declaration of the variable causes behavior changes in its scope before the line in which it is declared.
4. The side effects of a declaration are produced before evaluating the rest of the code that contains it.

The four function declarations above are hoisted with type 1 behavior; `var` declaration is hoisted with type 2 behavior; `let`, `const`, and `class` declarations (also collectively called *lexical declarations*) are hoisted with type 3 behavior; `import` declarations are hoisted with type 1 and type 4 behavior.