Homework #4

	Scope (global, function / locally, block)	Initial value when hoisted	can be redeclared (yes or NO)	can be updated (yes or No)
Var	Function Scope	undefined	yes	yes
Const	Block Scope	Reference Error	NO	NO
Let	Block Scope	Reference error	NO	yes

① Var

Output

```
undefined
undefined
```

This is hoisting, because of that it will pull var a, b, to the top of the execution of program variable will be undefined. After hoisting it becomes.

```
var a, b;  // undefined
function test() {
        console.log(a);   } // Print · undefined
        console.log(b);   }
test();
var a=1, b=2;
```

② Let

output

Reference error: Cannot access 'a' before initialization
This is hoisting, in case of let when variable are hosted it is not initialized with a value of undefined. rather they are in state called Temperal Dead Zone and they are not initialized until their definition are evaluated.

③ Const

Output

Reference error: cannot access 'a' before initialization. This is hoisting in case of const when variable are hosted it is not initialized with a value of undefined, rather they are in State called Temperal Dead Zone and they are not initialized until their definition are evaluated.

Var, Const, and Let.

① Output

100
2

This is scope, console.log(a) will give 100 it will take value of var a which is 100, it will check if in its current scope value of a variable is defined or not, which is their var a = 100 and will print 100.
console.log(b) - this will print 2 because let value is only available till inner scope and we can't access it outside as let is block scope. So it will take the value of const b=2 from outer scope and ~~it~~ will print the values as 2.

② Reference error: c is not defined.
③ This is because const is block scoped so it can be accessed only within the if(true){} and it can't be accessed outside. So it will give reference error.