Var and Let

- In ES6 we have 3 ways to declare variables.

```
var a = 10;
let b = 20;
const c = 30;
```

- The type of declaration depends on the scope of the variable.
- Scope is a fundamental concept in all programming languages that defines the usage and visibility of variables.

var & let

- Unlike var keyword, which is global scope,or locally to an entire function reguardless of block, let allows you to define a variable that is limited to the block it is defined in.

```
if (true) {
  let a = 10;
}
alert(a); // error
```

- In this case the variable is defined using block scope (let) and is being used in the outer block.
- Hence the variable is not accessible outside the block.
- To demonstrate the diffrence between var and let, let is used in the following example.

```
function test() {
  var a = 10;
  if (true) {
    let b = 20;
    console.log(a); // 10
    console.log(b); // 20
  }
  console.log(a); // 10
  console.log(b); // error
}
```

- One of the best uses of let is in loops.

```
for (let i = 0; i < 10; i++) {
  console.log(i); // 0,1,2,3,4,5,6,7,8,9
}
console.log(i); // error
```

- Since the variable is defined using block scope, it is not accessible outside the block.
- This behaviour is as it is supposed to be for loops as the iterator variable is not accessible outside the loop, and isnt supposed to be accessible outside the loop.

> Note : let is not subject to Variable Hoisting, which means that let declarations are not hoisted to the top of the scope or the current execution context.

Quiz Time

- What will be the output of the following code?

```
function func() {
  let a = 10;
  if (true) {
    let a = 20;
    console.log(a);
  }
  console.log(a);
}
func();
```

Answer: 20,10

const keyword

- const variables are like let variables, and have same scope as let, but they cannot be reassigned.
- const variables are immutable i.e. they are not allowed to change their value (reassign).

```
const a = 10;
a = 20; // error
```

- Simmilarly const declarations are not hoisted to the top of the scope or the current execution context.
- Also note that ES6 code will only run in supported browsers, and older browsers will not run ES6 code and will throw an error(syntax error).

Quick Quiz

- Fill in the blanks for appropriate keywords.

```
____ length = 10;
let sum = 0;
for(___ i = 0; i < length; i++) {
  sum += i;
}
```