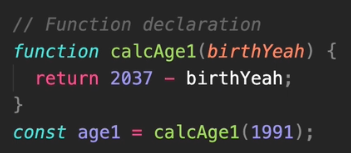
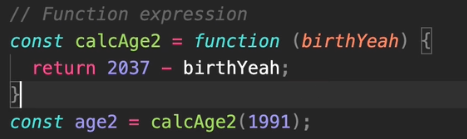
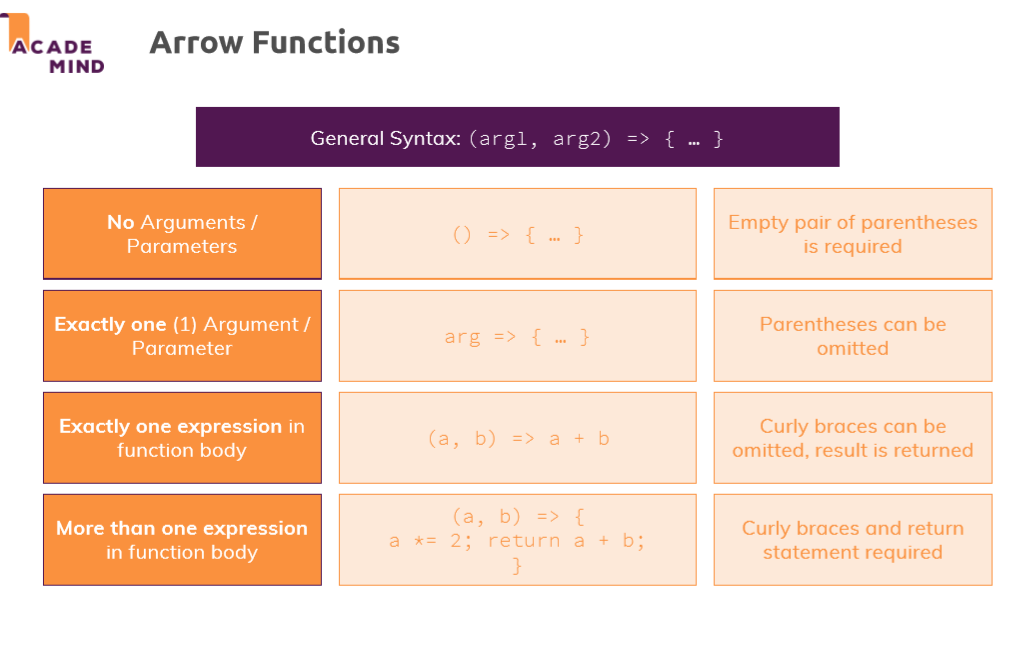
There is two way functions are written in javascript.

* **Function Declaration:** this type of function is hoisted (raised up to the top of the code). So, order is not important here. ( wont use this myself personally )
* **Function Expression:** This is not hoisted. So, order is important here since the code will execute from top to bottom. 
* **Arrow Function:** ES6 version of function.
  1. It doesn’t have access to **this** or **super.** If we use this in arrow function, it will get the parent context.
  2. Shouldn’t be used as **methods** or **constructor**.
  3. Doesn’t have **arguments or new.target** keyword.
  4. Can’t use **yield,** within its body.
  5. Not suitable for call, apply and bind methods, which generally rely on establishing a scope.



* In JavaScript there is no passing value by reference, when we pass an object as a parameter, we send a different variable but both point at same memory location. So, any changes made inside the function, reflects in both variable.
* But in primitive type variables, we actually pass a copy of the variable, so any changes made inside the function, doesn’t get reflected on the actual variable.

