// regular function:

```
func functionName (parameter: parameterType) -> returnType {
    //Does something
    return output
}
```

// convert func to closure

```
func sum (firstNumber: Int, secondNumber: Int) -> Int {
    return firstNumber + secondNumber
}
```

⇓

```
{ (firstNumber: Int, secondNumber: Int) -> Int in
    return firstNumber + secondNumber
}
```

// can pass a function to another function and return another function

```
import UIKit

func calculator (n1: Int, n2: Int, operation: (Int, Int) -> Int) -> Int {
    return operation (n1, n2)
}
```

```
func multiply (no1: Int, no2: Int) -> Int {
    return no1 + no2
}
calculator (n1: 2, n2: 3, operation: multiply)
```

⇒

```
calculator (n1: 2, n2: 3, operation: { (no1: Int, no2: Int) -> Int in
    return no1 * no2
})
```

⇓ data referal for input and return

```
calculator (n1: 2, n2: 3, operation: { (no1, no2) in
    return no1 * no2
})
```

⇓ inside closure only one line

```
calculator (n1: 2, n2: 3, operation: { (no1, no2) in no1 * no2 })
```

⇓ $0 for first paramater, $1 for second one

```
calculator (n1: 2, n2: 3, operation: { $0 * $1 })
```

⇓ trailing closure: if the last parameter is a func

```
calculator (n1: 2, n2: 3) { $0 * $1 }
```

// example

```
import UIKit

let array = [1, 2, 3, 4, 5]

func addOne (n1: Int) -> Int {
    return n1 + 1
}
array.map (addOne)
```

⇒ `array.map ({ $0 + 1 })` ⇒ (trailing) `array.map { $0 + 1 }`

```
// convet to string
let newArray = array.map { "\($0)" }
```

```
// call back functions
import UIKit

class Firebase {
    func creatUser (username: String , password : String, completion : (Bool, Int) -> void){
        // do something time consuming
        var isSuccess = true
        var userID = 123
        completion (isSuccess, userID)
    }
}

class MyApp{
    func registerButtonPressed (){
        let firebase = Firebase()
        firebase . createUser (name : "A", password : "1234", completion : completed)
    }

    func completed (isSuccess : Bool, userID : Int){
        print ("registeration is succesful : \(isSucess)")
        print ("userID is \(userID)")

    }
}
let myApp = MyApp()
myApp . registerButtonPressed()
```