Functions

No parameters, return nothing

```
func doNothing() {
    // crickets.mp3
}
doNothing()
```

One parameter, return nothing

```
func printSomething(times: Int) {
    for _ in 0..<times {
       print("'something'")
    }
}
printSomething(3)
```

Take int, return int

```
func multiplyNumberByItself(numberToMultiply
number: Int) -> Int {
    return number * number
}
let multipliedNumber =
multiplyNumberByItself(numberToMultiply: 3)
print(multipliedNumber)
```

No parameters, return int

```
func justReturnOne() -> Int {
    return 1
}
let justOne = justReturnOne()
print(justOne)
```

take two parameters, return tuple

```swift
func dontKnow(number number: Int, text:
String) -> (Int, String) {
    print("Number: \(number), text: \
(text)")

    return (number + 1, text + " asdf")
}
let tuple = dontKnow(number: 2, text:
"qwer")
print(tuple.0, tuple.1)

Take modifiable parameter, modify, return nothing
func asdf(inout number: Int) {
    number += 1
}

var number = 3
asdf(&number)
print(number)


                    Closures
//let helloPrinter = {
//  var hello = "Hello"
//  print(hello)
//}

let helloPrinter: () -> () = {
    var hello = "Hello"
    print(hello)
}
helloPrinter()

typealias closureIntInt = (Int) -> (Int)
```

```
let multiplyByTwoClosure: closureIntInt =
{ number -> Int in
    return number * 2
}
multiplyByTwoClosure(3)

func multiplyNumberByTwoFunction(number
number: Int, closure: closureIntInt) -> Int
{
    return closure(number)
}

multiplyNumberByTwoFunction(number: 10,
closure: multiplyByTwoClosure)
```