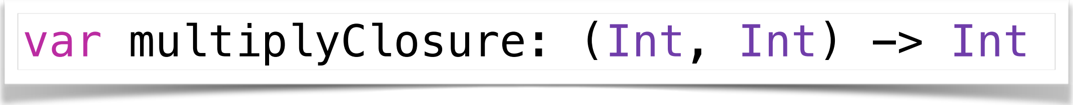
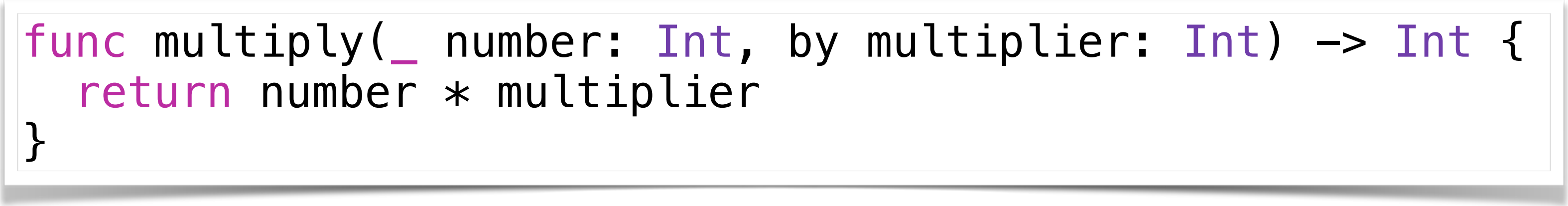
# INTERMEDIATE SWIFT 3

## Part #2: Closures

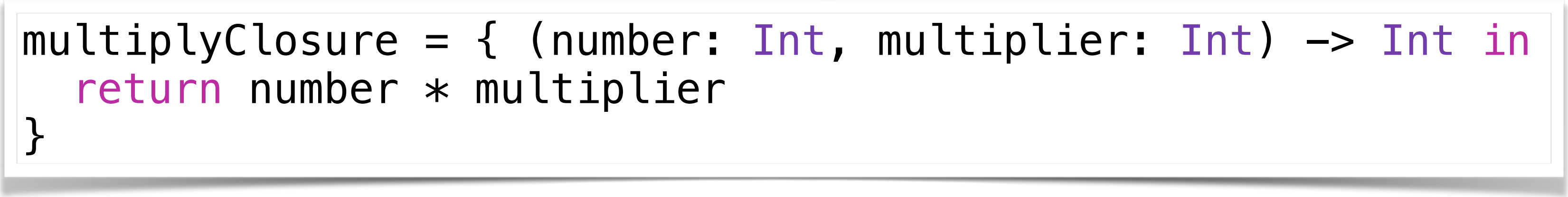
# Closures



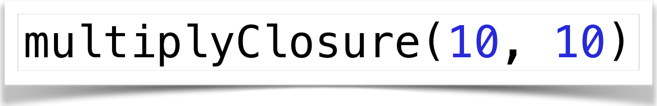
```
var multiplyClosure: (Int, Int) -> Int
```



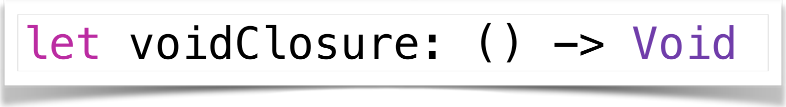
```
func multiply(_ number: Int, by multiplier: Int) -> Int {
  return number * multiplier
}
```



```
multiplyClosure = { (number: Int, multiplier: Int) -> Int in
  return number * multiplier
}
```



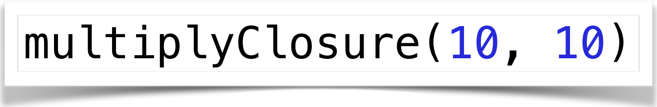
```
multiplyClosure(10, 10)
```



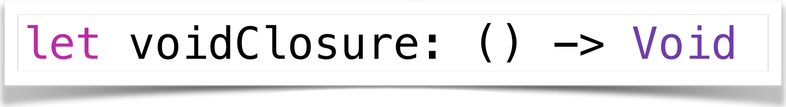
```
let voidClosure: () -> Void
```

# Closures

```
multiplyClosure = {
      ( number: Int, multiplier: Int)  -> Int
      in
          return number * multiplier
}
```

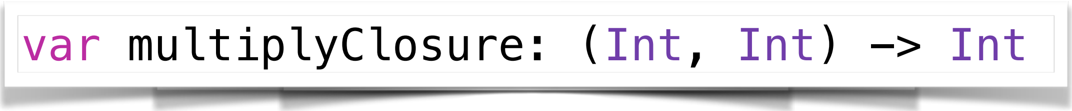


```
multiplyClosure(10, 10)
```

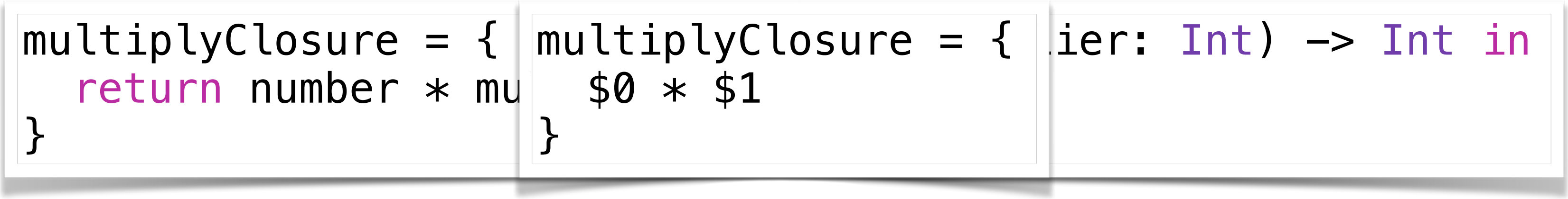


```
let voidClosure: () -> Void
```

# Type Inference



```
var multiplyClosure: (Int, Int) -> Int
```

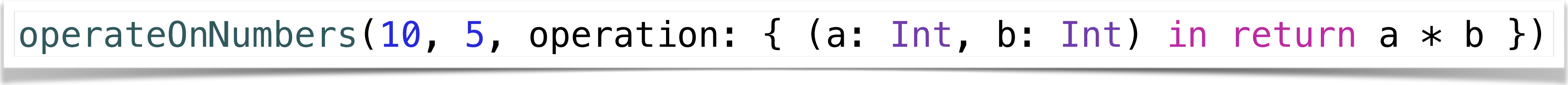


```
multiplyClosure = {
  return number * mu
}
```

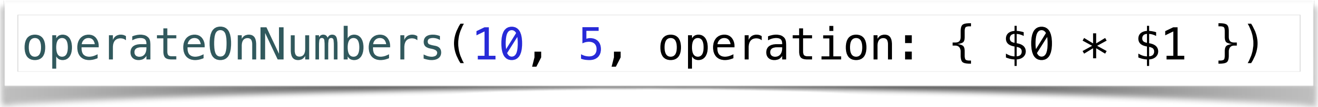
```
multiplyClosure = {
  $0 * $1
}
```

```
ier: Int) -> Int in
```

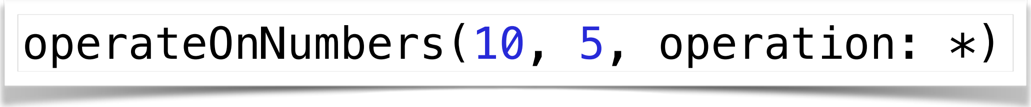
# Condensing Closures



```
operateOnNumbers(10, 5, operation: { (a: Int, b: Int) in return a * b })
```

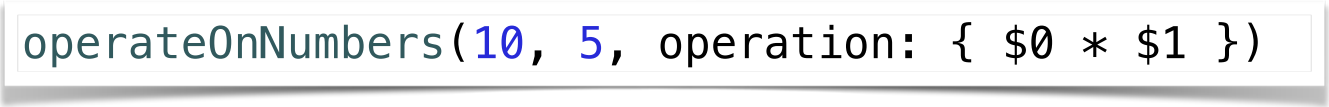


```
operateOnNumbers(10, 5, operation: { $0 * $1 })
```

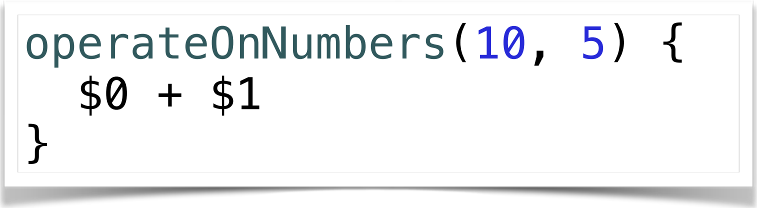


```
operateOnNumbers(10, 5, operation: *)
```

# Trailing Closures

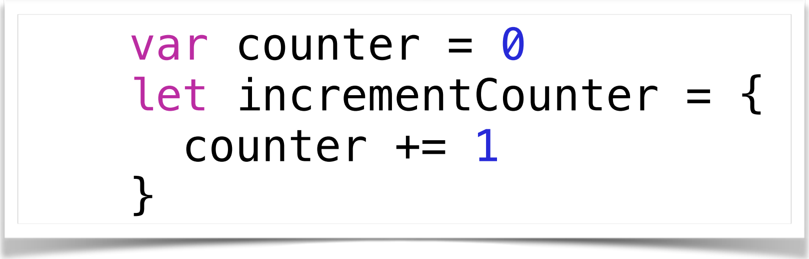


```
operateOnNumbers(10, 5, operation: { $0 * $1 })
```



```
operateOnNumbers(10, 5) {
  $0 + $1
}
```

# Closures



```
var counter = 0
let incrementCounter = {
  counter += 1
}
```

# Demo

# Challenge Time!