Lakehead UNIVERSITY | Teaching Commons

Mobile Computing Technology

February 11, 2025

Learning Objective

- Explain what a nullable variable is
- Several forms of iterations
- Accessing Arrays

More of

KOTLIN

Nullability

- refers to the ability of variables to have an absence of value
- Null is the absence of a value.
  - In C, for some data types, it means a 0
  - What is an empty string in C? Null?
- val name = “Ali”
- Val name = “”
- val name = null

Null

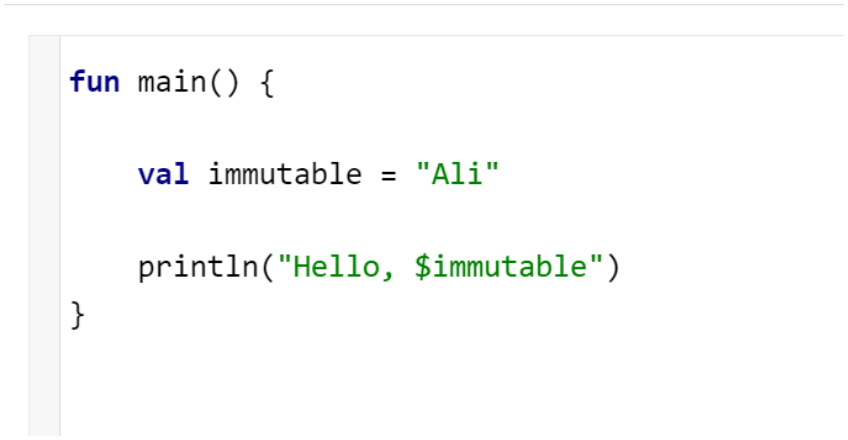
- In Kotlin, you can use null to indicate that there's no value associated with a variable.

Null

- In Kotlin, you can use null to indicate that there's no value associated with a variable.

Immutable variables

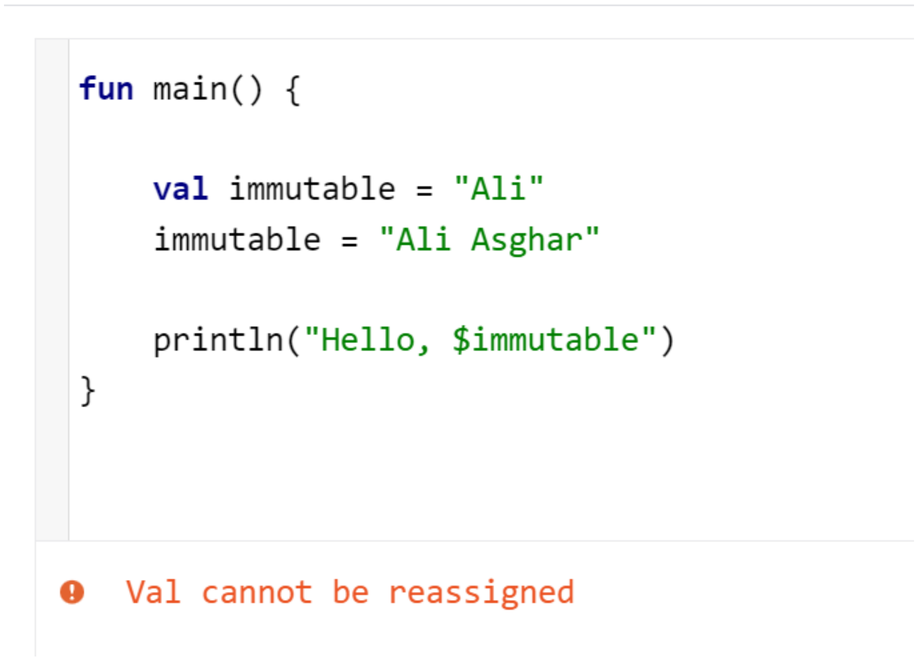
- Try this in Kotlin Playground



```kotlin
fun main() {

    val immutable = "Ali"

    println("Hello, $immutable")
}
```

Immutable variables

- Try this in Kotlin Playground



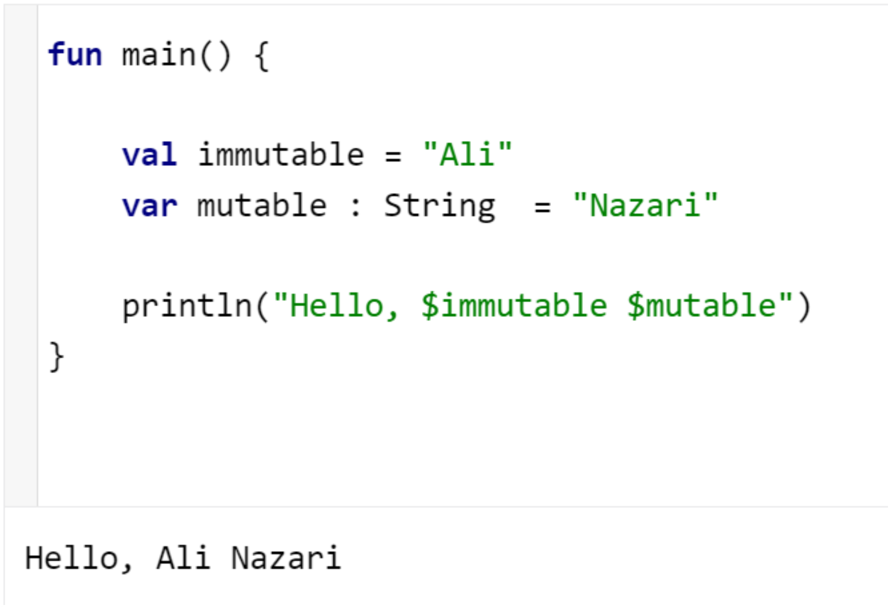
```
fun main() {

    val immutable = "Ali"
    immutable = "Ali Asghar"

    println("Hello, $immutable")
}
```

```
Val cannot be reassigned
```

Mutable variables

- Try this in Kotlin Playground

- Note the keyword we used for declaration
  - *val*
  - *var*

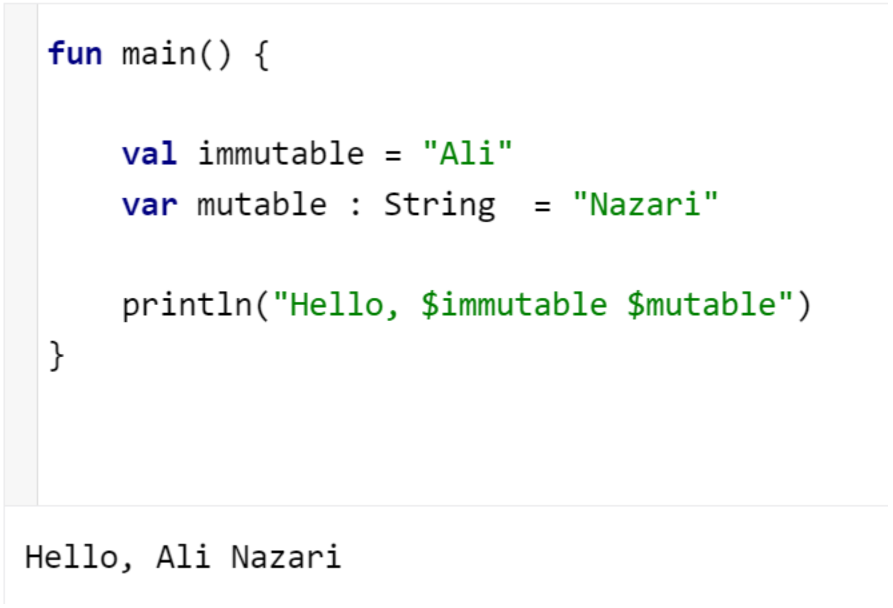


```kotlin
fun main() {

    val immutable = "Ali"
    var mutable : String  = "Nazari"

    println("Hello, $immutable $mutable")
}
```

```
Hello, Ali Nazari
```

Mutable variables

- Try this in Kotlin Playground
- Note the keyword we used for declaration
  - *val*
  - *var*

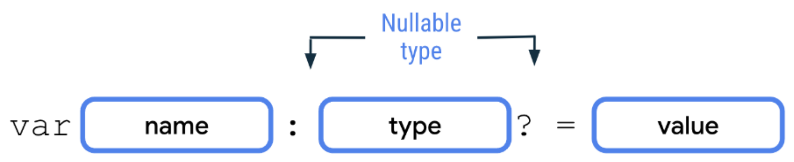


```
fun main() {

    val immutable = "Ali"
    var mutable : String  = "Nazari"

    println("Hello, $immutable $mutable")
}
```

```
Hello, Ali Nazari
```

Nullable variables

- In Kotlin, there's a distinction between nullable and non-nullable types
- A type is only nullable if you explicitly let it hold *null*

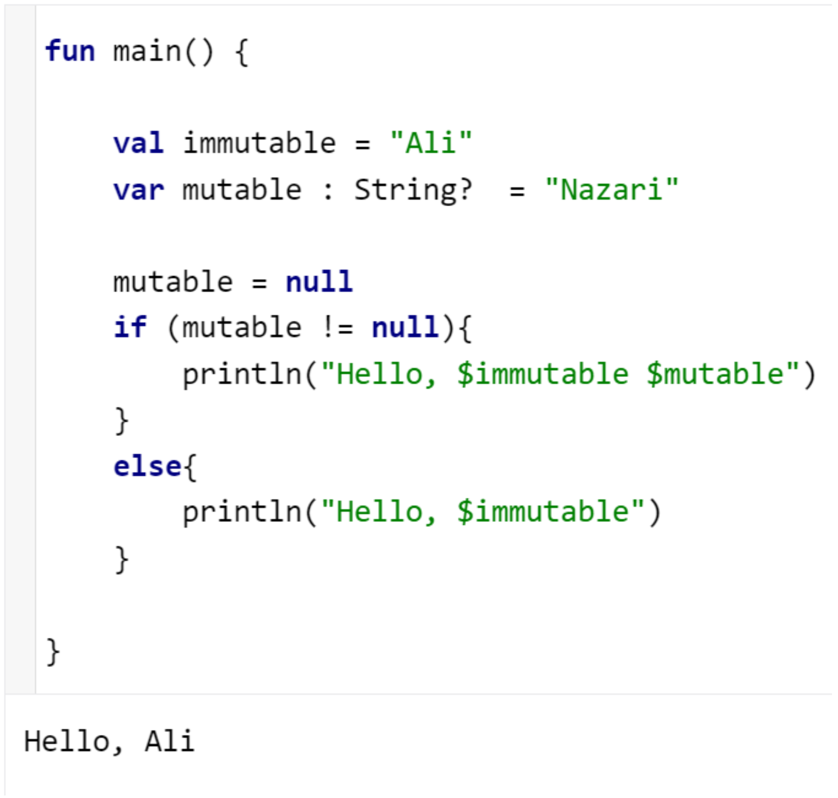


Nullable variables

To declare nullable variables in Kotlin, you need to add a ? operator to the end of the type

These are two different types

- String
- String?



```kotlin
fun main() {

    val immutable = "Ali"
    var mutable : String?  = "Nazari"

    mutable = null
    if (mutable != null){
        println("Hello, $immutable $mutable")
    }
    else{
        println("Hello, $immutable")
    }

}
```

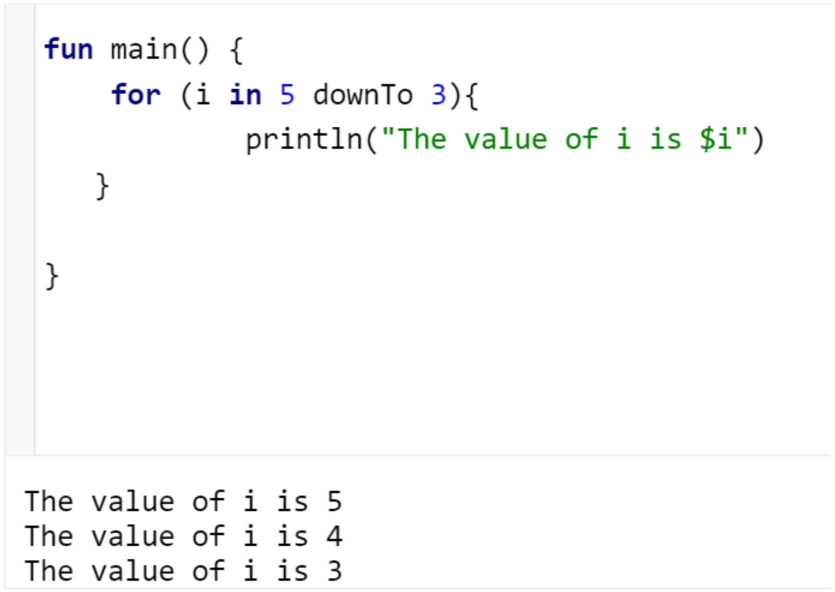
```
Hello, Ali
```

Variable Declaration

```
fun main() {

   val immutable = "Ali"
   var mutable : String?  = "Nazari"

   mutable = null
               if (mutable != null){
               println("Hello, $immutable $mutable")
   }
   else{
      println("Hello, $immutable")
   }

   // Can you declare variables anywhere
   // in the code or just at the beginning of a block?
   // Check it out yourself

}
```

Iteration

- Several ways for iteration
- Assess the following code
- What will be the output of println()

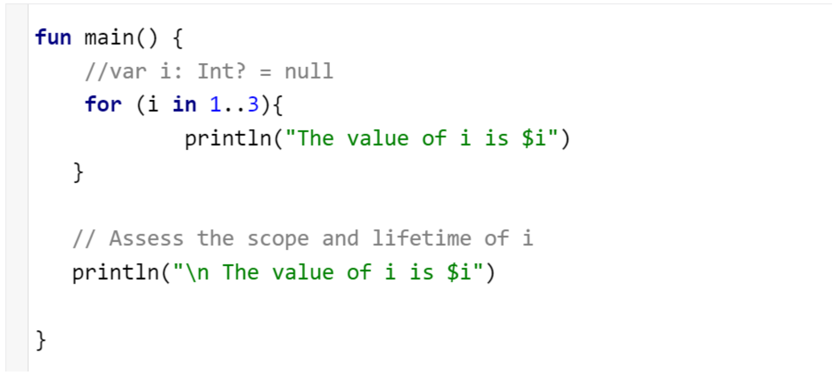


```
fun main() {
    for (i in 5 downTo 3){
            println("The value of i is $i")
   }

}
```

```
The value of i is 5
The value of i is 4
The value of i is 3
```

Iteration

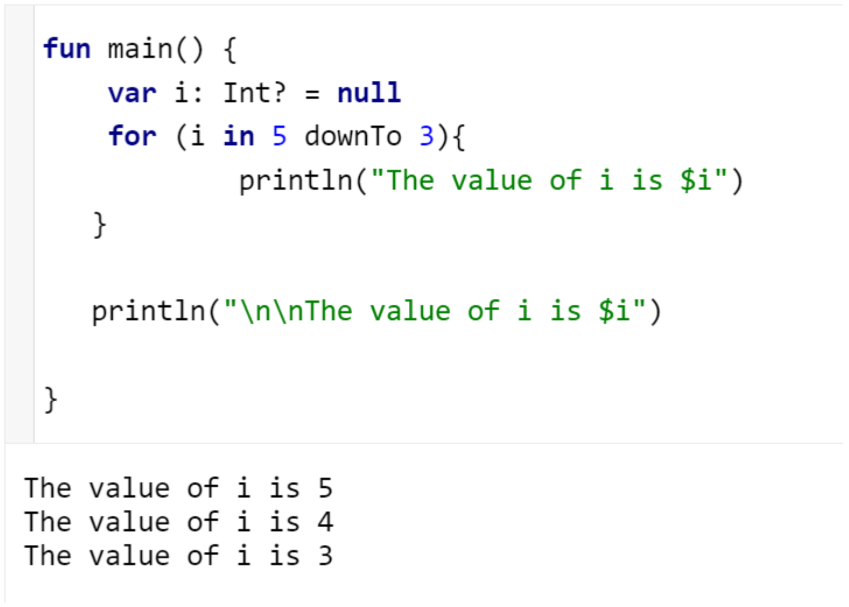
- Assess the following code
- What will be the output of the last println()



```
fun main() {
    //var i: Int? = null
    for (i in 1..3){
            println("The value of i is $i")
   }

   // Assess the scope and lifetime of i
   println("\n The value of i is $i")

}
```

Iteration

- Assess the following code
- What will be the output of the last println()?

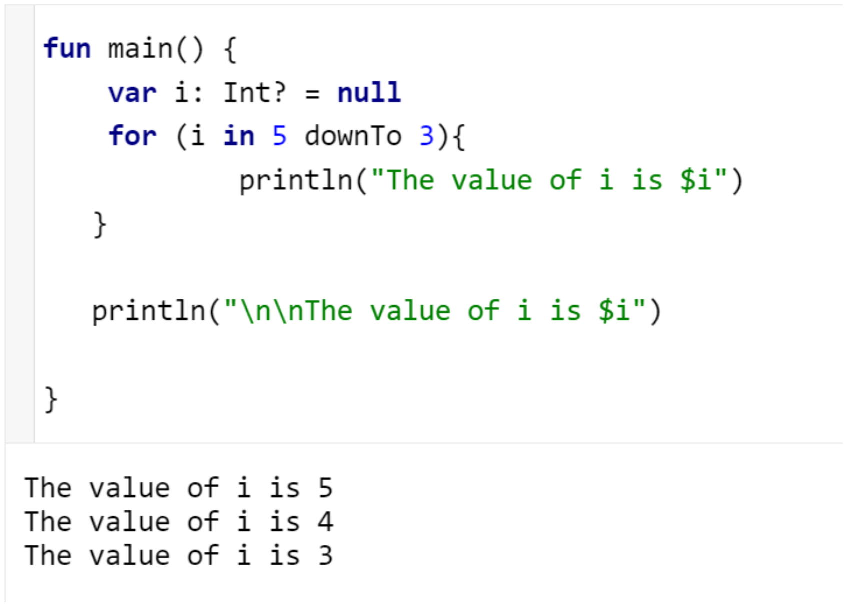


```
fun main() {
    var i: Int? = null
    for (i in 5 downTo 3){
            println("The value of i is $i")
   }

   println("\n\nThe value of i is $i")

}
```

```
The value of i is 5
The value of i is 4
The value of i is 3
```

Iteration

- Assess the following code
- What will be the output of the last println()?

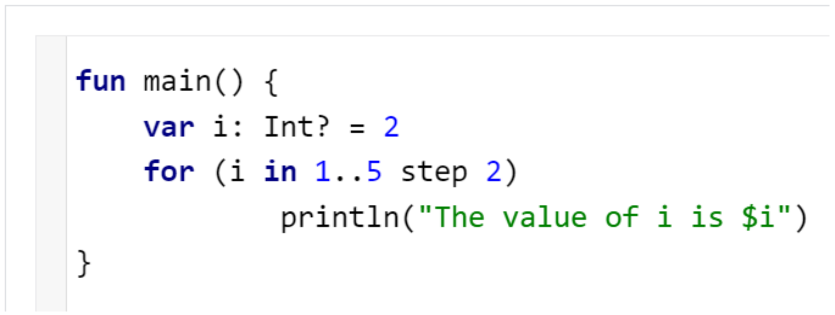


```
fun main() {
    var i: Int? = null
    for (i in 5 downTo 3){
            println("The value of i is $i")
   }

   println("\n\nThe value of i is $i")

}
```

```
The value of i is 5
The value of i is 4
The value of i is 3
```

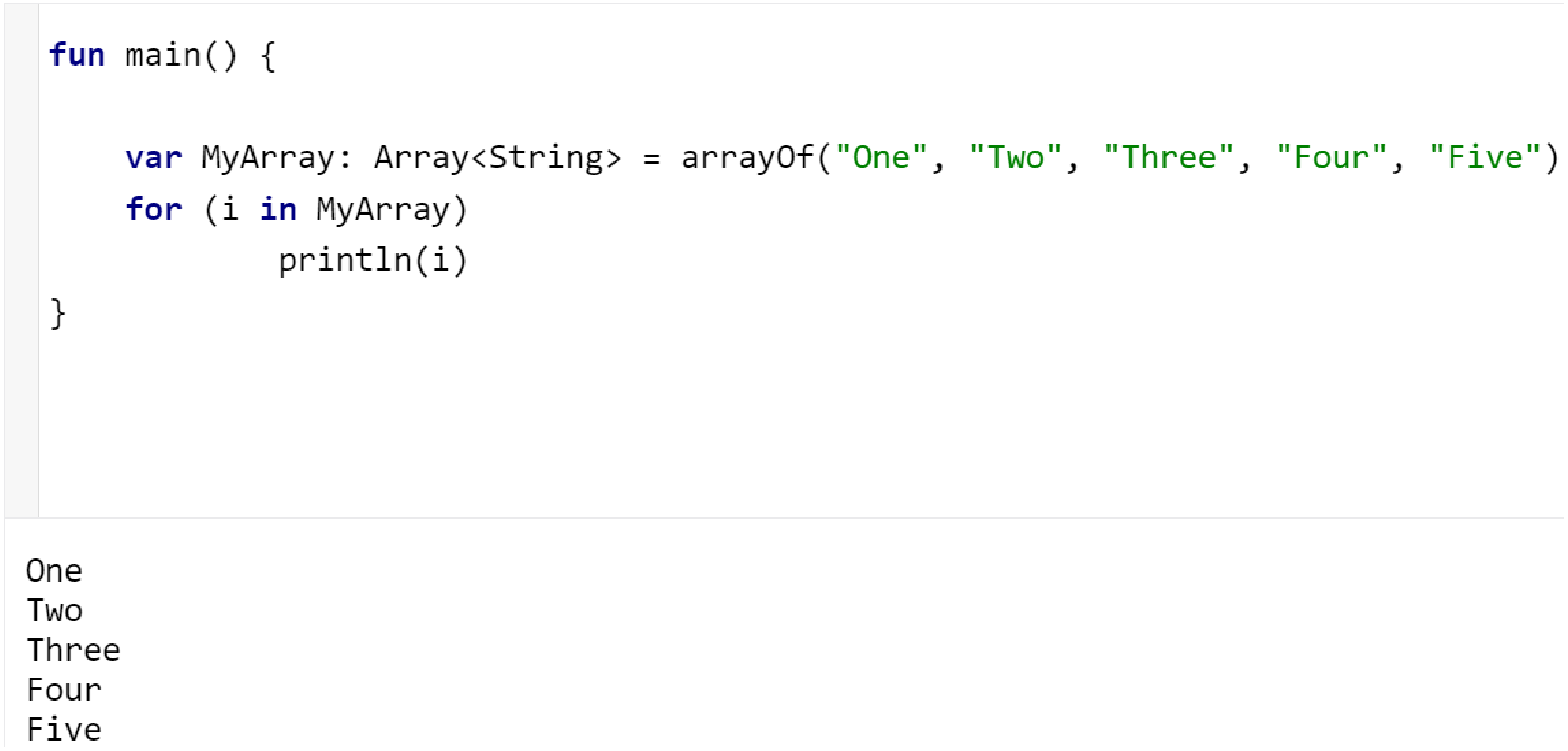
Iteration

- Assess the following code
- What will be the output of the last println()?



```
fun main() {
    var i: Int? = 2
    for (i in 1..5 step 2)
            println("The value of i is $i")
}
```

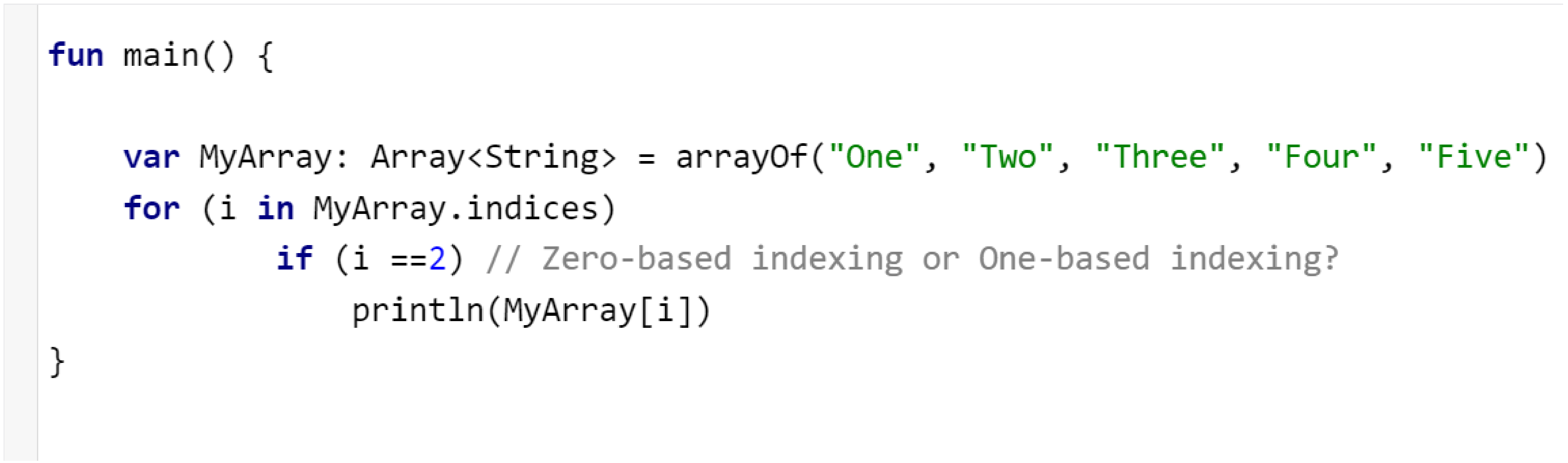
Iteration



```kotlin
fun main() {

    var MyArray: Array<String> = arrayOf("One", "Two", "Three", "Four", "Five")
    for (i in MyArray)
            println(i)
}
```

```
One
Two
Three
Four
Five
```

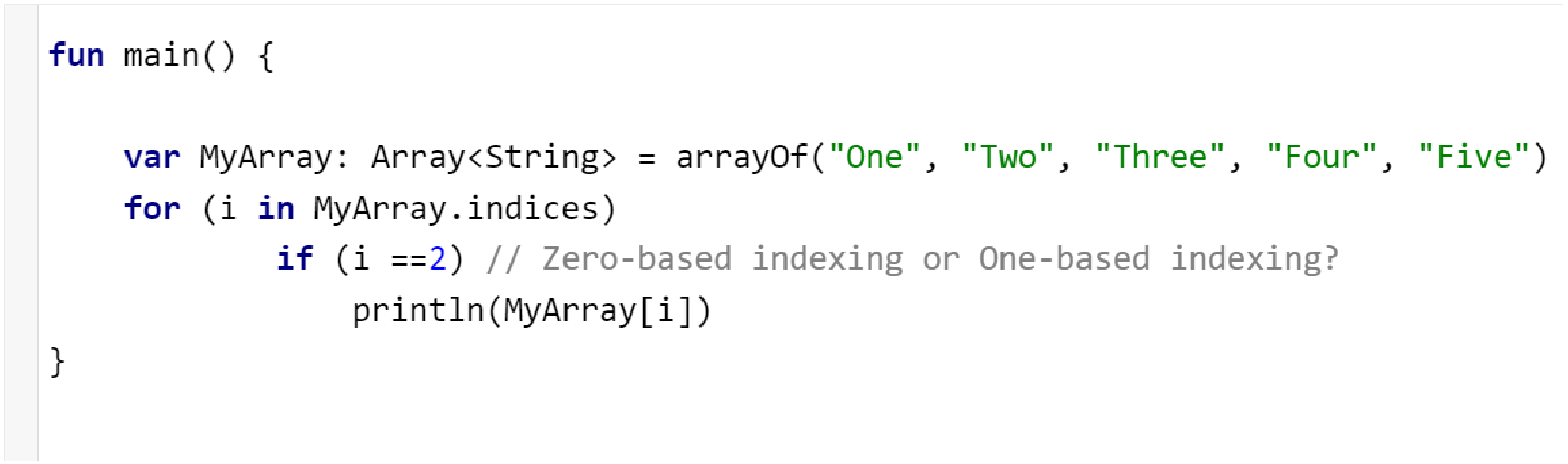
Iteration



```
fun main() {

    var MyArray: Array<String> = arrayOf("One", "Two", "Three", "Four", "Five")
    for (i in MyArray.indices)
            if (i ==2) // Zero-based indexing or One-based indexing?
                println(MyArray[i])
}
```

- Try to print the second element only

Iteration

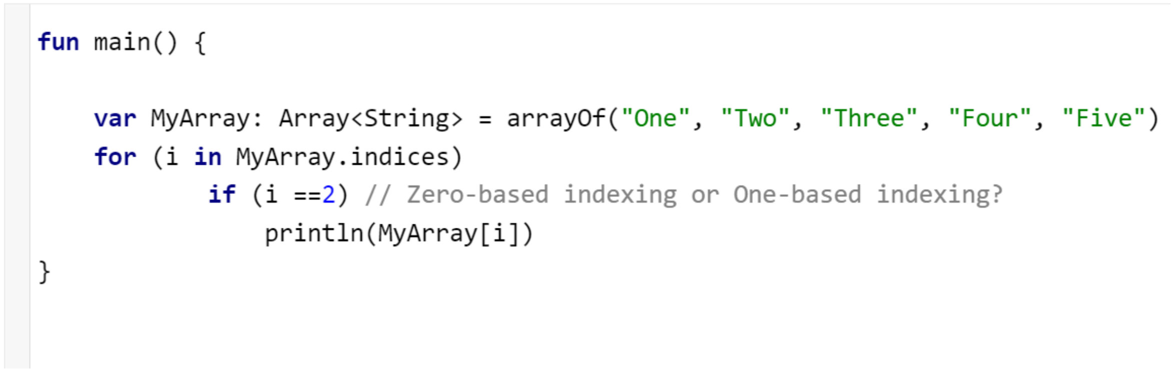


```
fun main() {

    var MyArray: Array<String> = arrayOf("One", "Two", "Three", "Four", "Five")
    for (i in MyArray.indices)
            if (i ==2) // Zero-based indexing or One-based indexing?
                println(MyArray[i])
}
```

- Try indexing at zero

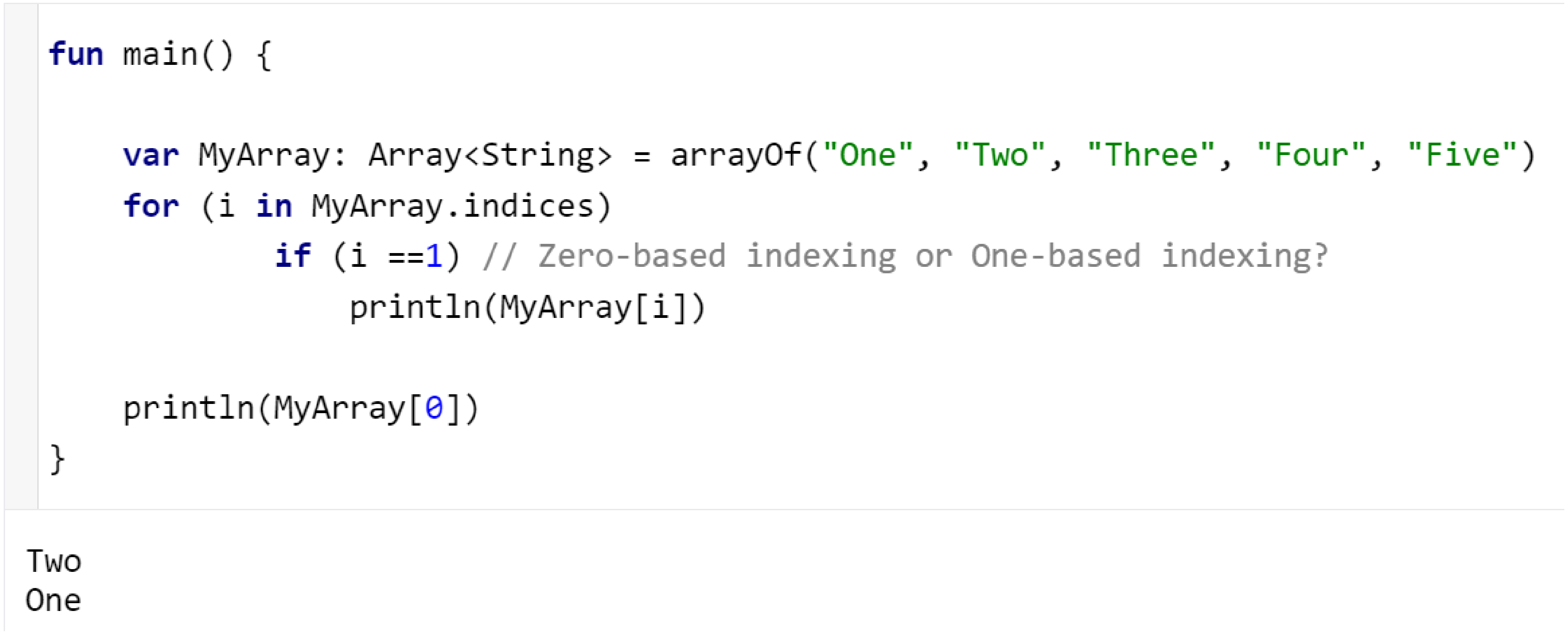
Iteration

- Does Kotlin support range checking?
- Can we used negative values for the index range?
- Try negative indexing, e.g. MyArray[−1]



```
fun main() {

    var MyArray: Array<String> = arrayOf("One", "Two", "Three", "Four", "Five")
    for (i in MyArray.indices)
            if (i ==2) // Zero-based indexing or One-based indexing?
                println(MyArray[i])
}
```

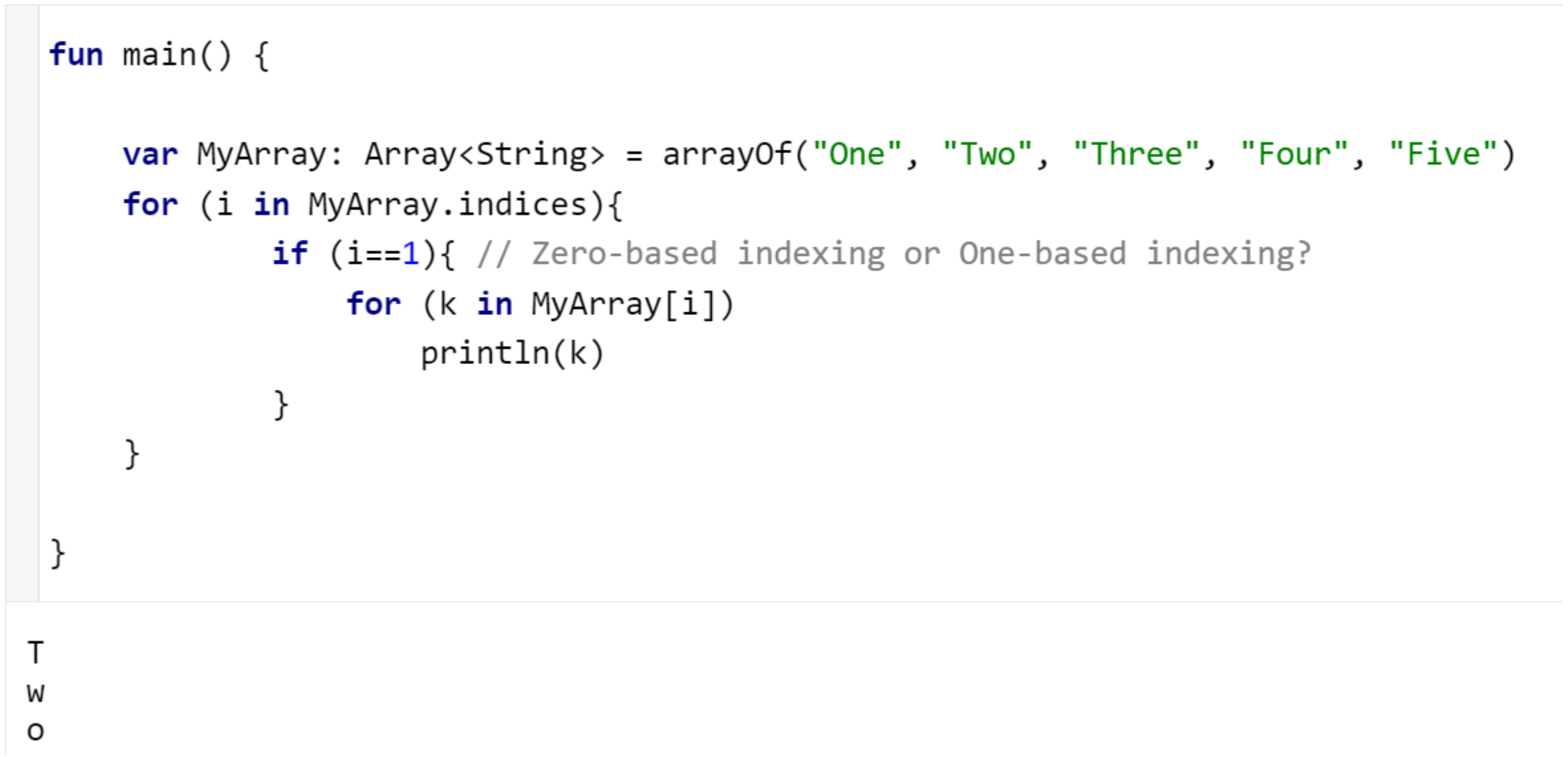
Iteration



```
fun main() {

    var MyArray: Array<String> = arrayOf("One", "Two", "Three", "Four", "Five")
    for (i in MyArray.indices)
            if (i ==1) // Zero-based indexing or One-based indexing?
                println(MyArray[i])

    println(MyArray[0])
}
```

```
Two
One
```

Iteration



```
fun main() {

    var MyArray: Array<String> = arrayOf("One", "Two", "Three", "Four", "Five")
    for (i in MyArray.indices){
            if (i==1){ // Zero-based indexing or One-based indexing?
                for (k in MyArray[i])
                    println(k)
            }
    }

}
```

```
T
w
o
```

Summary