GET STARTED WITH

a KodeKloud Introductory Course

• ARRAYS SLICES MAPS

Arrays

An array is an ordered **collection of elements**, all of the same data type.

Array Initialization

main.go

```go
package main
import "fmt"

func main() {
  var fruits[4]string=
[4]string{"apples","pears", "oranges",
"lemons"}
  fmt.Println(fruits)
}
```

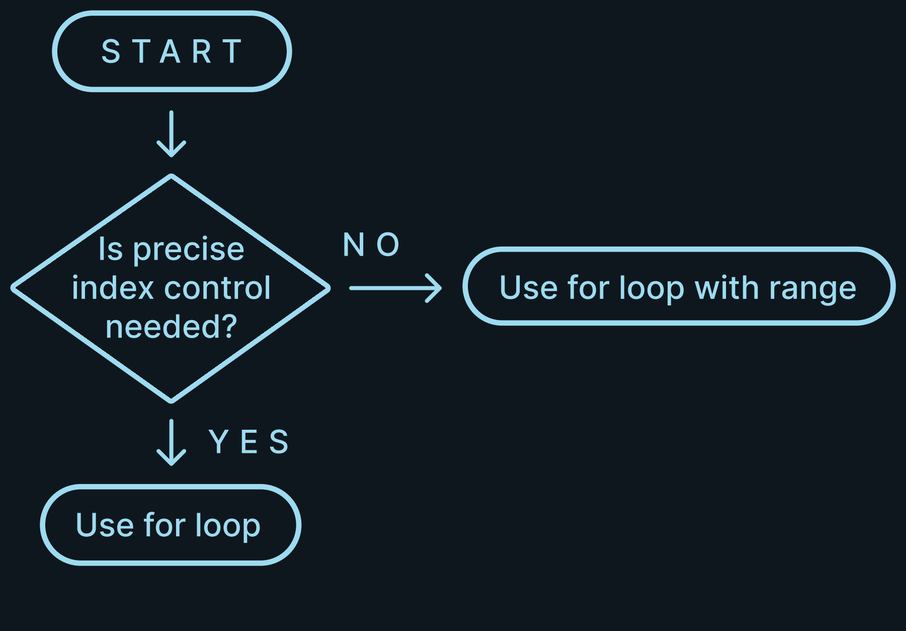
- Name of the array
- Length of the array
- Data type
- Values

! Arrays are fixed-size, once declared, the size of an array cannot be changed.

Looping Through An Array

Looping through an array involves systematically visiting **each element in the array**, one at a time, to perform some operation or gather information.

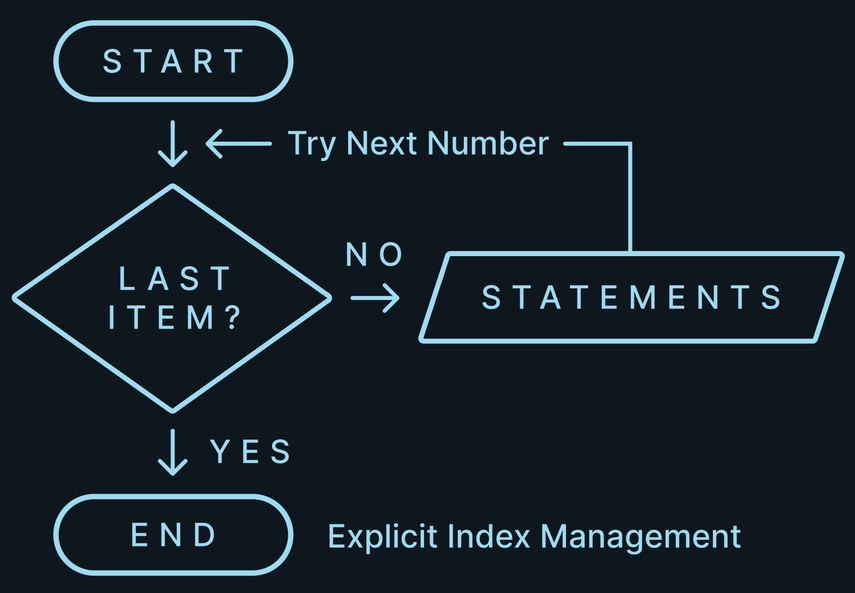
Decision Making Process



! When using range with slices or arrays, it returns both the index and the value. **If you only need the value, you can use an underscore (_) to discard the index.**

How To Loop

Using for loop



Explicit Index Management

For Loop Declaration

main.go

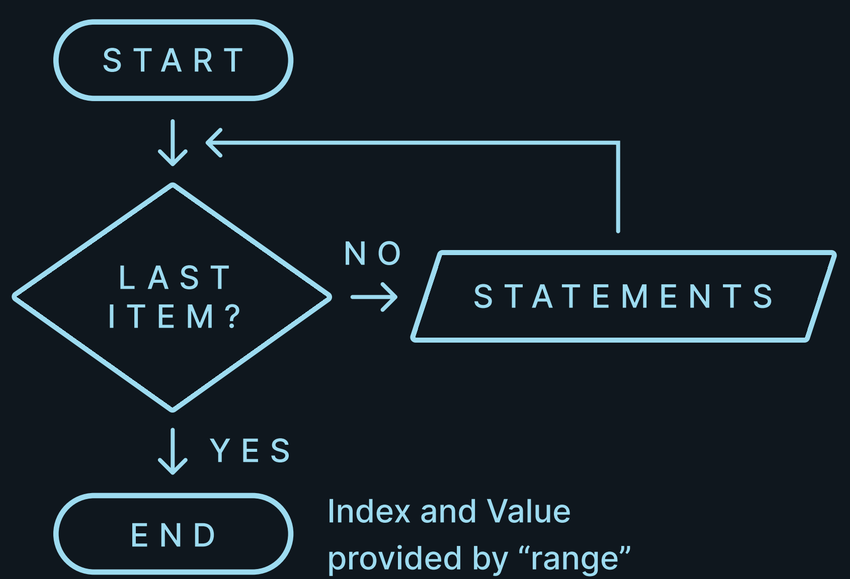
```go
package main
import "fmt"

func main() {
  var grades [5]int = [5]int{90, 80, 70, 80, 97}

  for i := 0; i < len(grades); i++ {
    fmt.Println(grades[i])
  }
}
```

- Initialization
- Post-Condition
- Condition
- Loop body

Using Range Keyword



Index and Value provided by “range”

Range Loop Declaration

main.go

```go
package main
import "fmt"

func main() {
  var grades [5]int = [5]int{90, 80, 70, 80, 97}
  for index, element := range grades {
    fmt.Println(index, "=>", element)
  }
}
```

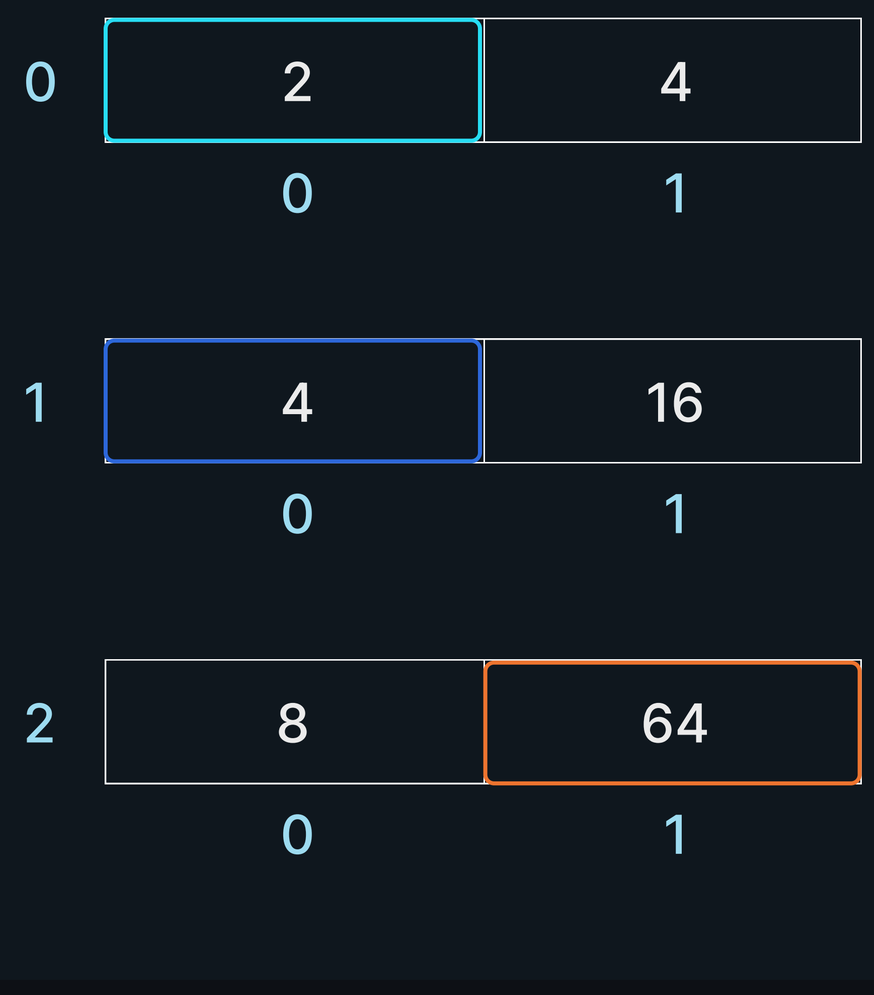
- Variables
- Loop body
- Rage expression

Multidimensional Arrays

A multidimensional array is **an array that has more than one dimension.** This concept allows you to organize data in a grid-like structure, such as a table or matrix.

2 Dimensional Arrays

2 DIMENSIONAL ARRAY




```
arr[2][1] => 64
arr[1][0] => 4
arr[0][0] => 2
```

main.go

```go
package main
import "fmt"

func main() {
  arr := [3][2]int{{2, 4}, {4, 16}, {8, 64}}
  fmt.Println(arr[2][1])
}
```

- Size of the Array
- Array Initialization

Be mindful of memory usage, especially when working with large multidimensional slices, as **dynamic resizing and allocation can impact memory efficiency.**

Go does not have native multidimensional arrays. Instead, **you use slices of slices** to create multidimensional data structures. Each slice represents a dimension.

KodeKloud

Ready to embark on your journey into the world of Go (Golang)?

Join our introductory course to Go (Golang) and unlock the power of this versatile programming language!

TAUGHT BY:

Priyanka Yadav

★★★★☆ 4.8

kode.wiki/48m805N