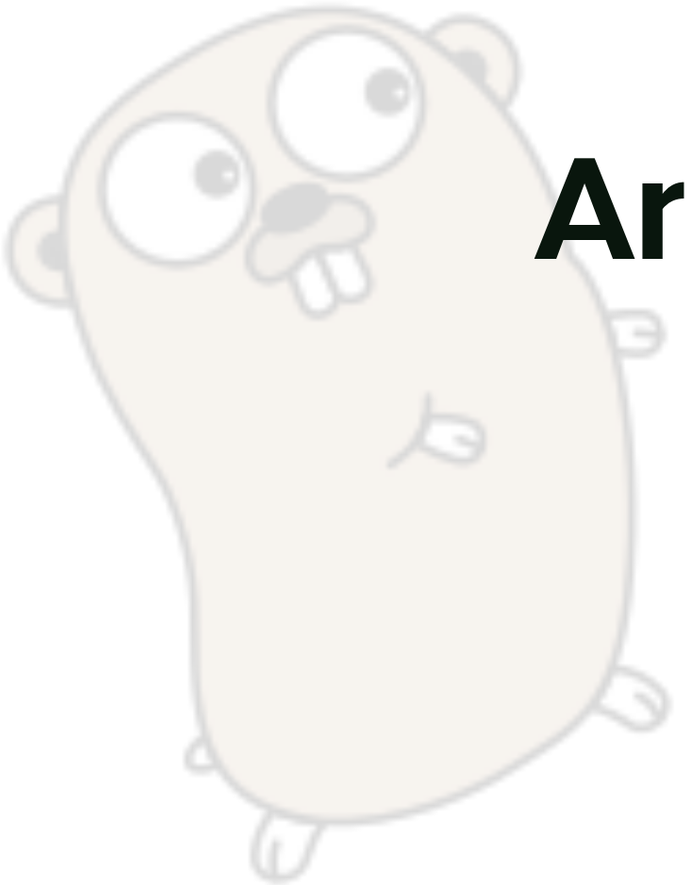
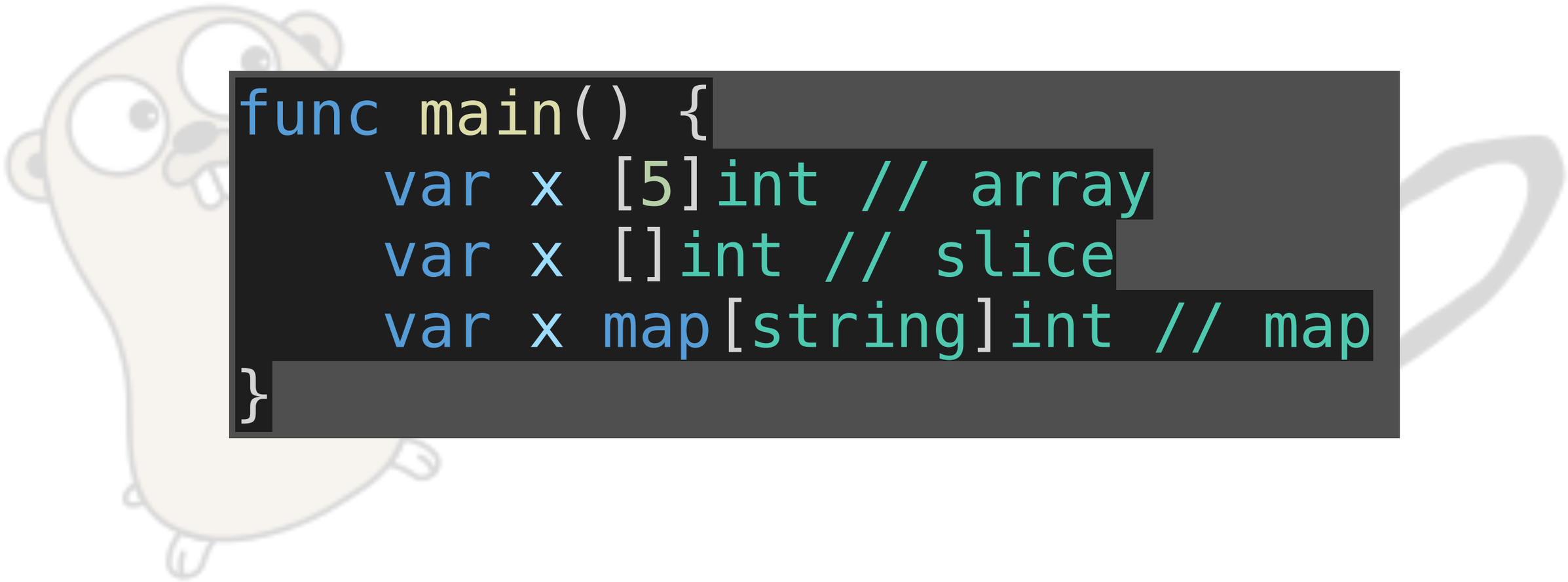
Collections:

Arrays, Slices, Map



golang : collections



```go
func main() {
    var x [5]int // array
    var x []int // slice
    var x map[string]int // map
}
```

golang : arrays

create main.go in folder chapter7-1 :

```go
package main

import "fmt"

func main() {
    var x [5]int
    x[3] = 4
    fmt.Println(x)

    x = [5]int{1, 2, 3, 4, 5}
    fmt.Println(x)

    y := [...]int{1, 2, 3, 4, 5, 6, 7, 8, 9, 0}
    fmt.Println(y)
}
```

run -> no error -> push to your git repository

golang : slice

create main.go in folder chapter7-2 :

```go
package main

import "fmt"

func main() {
    slice := make([]int, 3)
    slice[0] = 1
    slice[1] = 2
    slice[2] = 3

    fmt.Println(slice)

    slice2 := []int{1, 2, 3, 4, 5}
    fmt.Println(slice2)

    fmt.Println("Slice with length and capacity")
    fmt.Printf("slice: length %v, capacity %v, %v\n", len(slice), cap(slice), slice)

    // append
    for i := 4; i < 15; i++ {
        slice = append(slice, i)
    }
    fmt.Printf("slice: length %v, capacity %v, %v\n", len(slice), cap(slice), slice)
}
```

run -> no error -> push to your git repository

golang : slice

create main.go in folder chapter7-3 :

Create slice from array

```go
package main

import "fmt"

func main() {
    arr := [5]int{1, 2, 3, 4, 5}
    fmt.Println(arr)

    slice := arr[0:3]
    fmt.Println(slice)
}
```

run -> no error -> push to your git repository

golang : slice

create main.go in folder chapter7-4 :

Copy slices

```go
package main

import "fmt"

func main() {
    slice := []int{1, 2, 3}
    fmt.Println(slice)
    newSlice := make([]int, 2)
    fmt.Println(newSlice)
    copy(slice, newSlice)
    fmt.Printf("slice: %v\n", slice)
    fmt.Printf("slice: %v\n", newSlice)
}
```

run -> no error -> push to your git repository

golang : map

create main.go in folder chapter7-5 :

```go
package main

import "fmt"

func main() {
    var x map[string]int
    x = make(map[string]int)
    x["key"] = 10
    fmt.Println(x)
    fmt.Println(x["key"])

    y := map[string]int{
        "one":   1,
        "two":   2,
        "three": 3,
    }
    fmt.Println(y)
}
```

run -> no error -> push to your git repository

golang : map

create main.go in folder chapter7-6 :

Delete map

```go
package main

import "fmt"

func main() {
    x := map[string]int{
        "one":   1,
        "two":   2,
        "three": 3,
    }
    fmt.Println(x)

    delete(x, "two")
    fmt.Printf("After delete: %v\n", x)
}
```

run -> no error -> push to your git repository

golang : map

create main.go in folder chapter7-7 :

Avoid to check zero value

```go
package main

import "fmt"

func main() {
    mymap := make(map[int]int)
    mymap[1] = 1
    mymap[2] = 2

    fmt.Println(mymap[3])
    if mymap[3] != 0 {
        fmt.Println(mymap[3])
    }

    // ok?
    if value, ok := mymap[3]; ok {
        fmt.Println(value)
    }
}
```

run -> no error -> push to your git repository

golang : range and collections

create main.go in folder chapter7-8 :

Range: Array

```go
package main

import "fmt"

func main() {
    numbers := [5]int{1, 2, 3, 4, 5}
    for i := 0; i < len(numbers); i++ {
        fmt.Println(i, numbers[i])
    }
    fmt.Println("With Range")
    for i, number := range numbers {
        fmt.Println(i, number)
    }
}
```

run -> no error -> push to your git repository

golang : range and collections

create main.go in folder chapter7-9 :

Range: Slice

```go
package main

import "fmt"

func main() {
    slice := []int{1, 2, 3, 4, 5}
    for i, number := range slice {
        fmt.Println(i, number)
    }
}
```

run -> no error -> push to your git repository

golang : range and collections

create main.go in folder chapter7-10 :

Range: Map

```go
package main

import "fmt"

func main() {
    maps := map[string]int{
        "one":   1,
        "two":   2,
        "three": 3,
    }

    for key, number := range maps {
        fmt.Println(key, number)
    }
}
```

run -> no error -> push to your git repository

golang : range and collections

create main.go in folder chapter7-11 :

Range: String

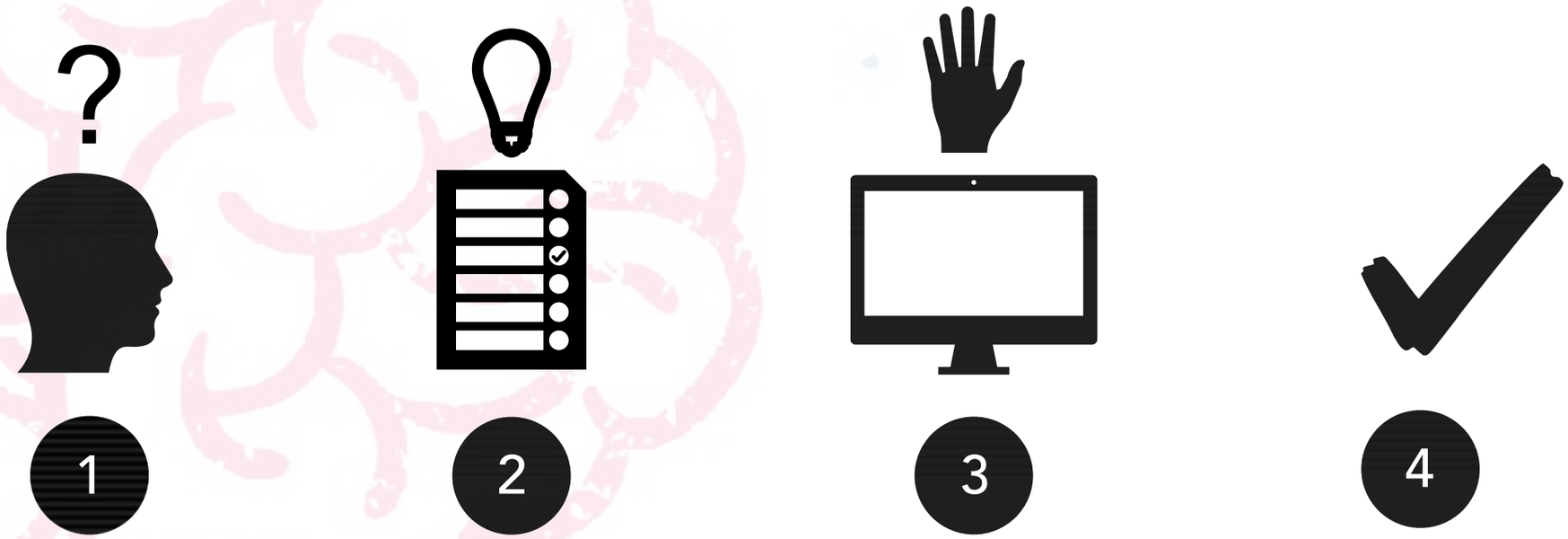
```
package main

import "fmt"

func main() {
    for i, c := range "golang" {
        fmt.Println(i, c)
        fmt.Printf("%v\n", string(c))
    }
}
```

run -> no error -> push to your git repository

Exercise

create exercise.go in folder chapter6-5 :

REFACTOR FIZZBUZZ ใน CHAPTER5-2
โดยใช้ TYPE ประเภท COLLECTION ของ GO
มาแทนที่เพื่อลด DUPLICATION ใน CODE



run -> no error -> push to your git repository