Golang Basic

Create New Project

```
mkdir <nama-project>
cd <nama-project>
go mod init <nama-project>

<-- Menjalankan GO -->
go run main.go
```

Type Data

Numerical Non - Decimal

Type Data	Range
uint8	0 ↔ 255
uint16	0 ↔ 65535
uint32	0 ↔ 4294967295
uint64	0 ↔ 18446744073709551615
uint	sama dengan uint32 atau uint64 tergantung nilai
byte	sama dengan uint8
int8	-128 ↔ 127
int16	-32768 ↔ 32757
int32	-2147483648 ↔ 2147483647
int64	-9223372036854775808 ↔ 9223372036854775807
int	sama dengan int32 atau int 64 (tergantung nilai)
rune	sama dengan int32

Numerical Decimal

```
var decimalNumber = 2.62

fmt.Printf("Bilangan decimal : %f\n", decimalNumber)
fmt.PrintF("Bilangan decimal : %.2f\n", decimalNumer)
```

Boolean

```
var exists bool = true
  fmt.Printf("exists? %t\n", exists)
```

String

```
say := `Nama Saya "Umar Sahid".
Salam kenal.
Mari Belajar "Golang".`

fmt.Println(say)
```

Variable

Variable Declaration

```
package main

import "imt"

func main(){
  var firstName string = "Umar"
  var lastName string
  lastName = "Sahid"

  fmt.Printf("halo %s %s!\n", firstName, lastName)
}
```

Declaration Variables without type data

```
var firstName string = "Umar"
lastName := "Sahid"

fmt.Printf("halo %s %s!\n", firstName, lastName)
```

Declaration Multi Varibales

```
var first, second, third string
first, second, third = "satu", "dua", "tiga"

seventh, eight, ninth := "tujuh", "delapan", "sembilan"
```

Declaration Underscore Varibales

```
_ = "Belajar Golang"
- = "Golang itu mudah"
name, _ := "Umar", "Sahid"
```

Constrants

```
const firstName string = "Umar"
fmt.Print("Halo ", firstName, "!\n")
```

Condition

```
var point = 8

if point = 10 {
  fmt.Println("lulus dengan nilai sempurna")
} else if point >= 5 {
  fmt.Println("lulus")
} else if point == 4 {
  fmt.Println("hampir lulus")
} else {
  fmt.Printf("tidak lulus. nilai anda %d\n", point)
}
```

Comparison Operator

Operator	Name	Description	Example
&&	Logical And	Return true if both statements are true	x < y && x > z
||	Logical Or	Return true if one of the statements is true	x < y || x > z
!!	Logical Not	Reverse the result return false if the result is true	!(x == y && x > z)

Switch Case

```
var point = 6

switch point {
case 8:
  fmt.Println("perfect")
case 7:
  fmt.Println("awesome")
default:
  fmt.Println("not bad")
}
```

Looping

For Range

```
var fruits =[4]string{"apple", "grape", "banana", "melon"}

for i, fruit := range fruits {
  fmt.Printf("elemen %d : %s\n", i, fruit)
}
```

For loop

```
for i :=0; i < 5; i++ {
  fmt.Println("Angka", i)
}
```

Loop Break Continue

```
for i := 0; i <= 10; i++ {
  if i % 2 == 1 {
    continue
  }

  if i > 8 {
    break
  }

  fmt.Println("Angka", i)
}
```

Function

Functions

```
func swap(x,y string)(string, string){
  return y,x
}

func main(){
  a, b := swap("hello", "world")
  fmt.Println(a,b)
}
```

Consecutive named function parameters

```
// from
func add(x, y int) int {
  return x + y
}

// to
func add(x, y int) int {
  return x + y
}
```

Multiple Return Values

```
func swap(x, y string)(string, string) {
  return y, x
}

func main() {
  a, b := swap("hello", "world")
  fmt.println(a, b)
}
```

Struct

Struct adalah kumpulan definisi variabel (atau property) dan atau fungsi (atau method), yang dibungkus sebagai tipe data baru dengan nama tertentu. Property dalam struct, tipe datanya bisa bervariasi. Mirip seperti **map**, hanya saja key-nya sudah didefinisikan di awal, dan tipe data tiap itemnya bisa berbeda.

Struct

```
type student struct {
  name string
  grade int
}

func main(){
  var s1 student
  s1.name = "Umar Sahid"
  s1.grade = 3

  fmt.Println("name : ", s1.name)
  fmt.Println("grade : ", s1.grade)
}
```

Embedded Struct

```
package main

import "fmt"

type person struct {
  name string
  age int
}

type student struct {
  grade int
  person
}

func main(){
  var s1 = student{}
  s1.name = "Umar Sahid"
  s1.age = 25
  s1.grade = 3

  fmt.Println("name : ", s1.name)
  fmt.Println("age : ", s1.age)
  fmt.Println("age : ", s1.person.age)
  fmt.Println("grade : " s1.grade)
}
```

Struct with Slice

```
type person struct {
  name string
  age int
}

var allStudents = []person{
  {name: "Ujang", age: 32},
  {name: "Bambang", age: 38},
  {name: "Indah", age: 22},
  {name: "Lestari" age: 25},
}

for _, student := range allStudents {
  fmt.Println(student.name, "age is", student.age)
}
```

Structure

Data Structure - Map

```
var chicken map[string]int
chicken = map[string]int{}

chicken["januari"] = 50
chicken["februari"] = 40

fmt.Println("januari", chicken["januari"]) // januari 50
fmt.Println("mei", chicken["mei"]) // mei 0
```

Data Structure - Array

```
var names [4]string
names[0] = "Trafalgar"
names[1] = "d"
names[2} = "water"
names[3] = "law"

fmt.Println(names[0], names[1], names[2], names[3])
```

Slice

Structure Slice

Slice di bagi menjadi 3 bagian yaitu pointer, length, capacity. Pointer ini index pertama dari array yang akan di slice. length adalah panjang array yang akan di slice. capacity adalah sisa dara dari pointer sampai index akhir array.

```
var fruits = []string{"apple", "grape", "banana", "melon"}
fmt.Println(fruits[0]) // apple
```

kode	output	Penjelasan
fruits[0:2]	[apple, grape]	semua elemen mulai dari index ke-0, hingga sebelum

		index ke-2
fruits[0:4]	[apple, grape, banana, melon]	semua elemen index ke-0, hingga sebelum index ke-4
fruits[0:0]	[]	menghasilkan slice kosong, karena tidak ada elemen sebelum index ke-0
fruits[4:4]	[]	menghasilkan slice kosong, karena tidak ada elemen yang dimulai dari index ke-4
fruits[4:0]	[]	error, pada penulisan fruits[a:b] nilai a harus lebih kecil atau sama dengan b
fruits[:]	[apple, grape, banana, melon]	semua elemen
fruits[2:]	[banana, melon]	semua elemen mulai index ke-2
fruits[:2]	[apple, grape]	semua elemen hingga sebelum index ke-2

```
var fruits = []string{"apple", "grape", "banana", "melon"}

var aFruits = fruits[0:3]
var bFruits = fruits[1:4]

var aaFruits = aFruits[1:2]
var baFruits = bFruits[0:1]

fmt.Println(fruits) // [apple grape banana melon]
fmt.Println(aFruits) // [apple grape banana]
fmt.Println(bFruits) // [grape banana melon]
fmt.Println(aaFruits) // [grape]
fmt.Println(baFruits) // [grape]

// Buah "grape" diubah menjadi "pinnaple"
baFruits[0] = "pinnaple"

fmt.Println(fruits) // [apple pinnaple banana melon]
fmt.Println(aFruits) // [apple pinnaple banana]
fmt.Println(bFruits) // [pinnaple banana melon]
fmt.Println(aaFruits) // [pinnaple]
fmt.Println(baFruits) // [pinnaple]
```

output	len()	cap()
[buah buah buah buah]	4	4
[buah buah buah ——]	3	4
—— [buah buah buah]	3	3

Standar Libraries (stdlib)

Most Used

1. bytes
2. crypto
3. errors
4. fmt
5. time
6. string
7. os
8. io
9. log
10. regexp
11. math
12. database
13. go

Miscellaneous

archive	index
bufio	mime
compress	net
container	path
debug	reflect
encoding	runtime
expvar	sort
flag	strconv

hash	sync
html	syscall
image	testing
unicode	text
unsafe	

Defer

```
package main

import "fmt"

func main(){
  defer fmt.Println("halo")
  fmt.Println("selamat datang")
}
```

Pointer

Variabel bertipe Pointer ditandai dengan adanya tanda asterisk (*) tepat sebelum penulisan tipe data ketika deklarasi

```
var number *int
var name *string
```

▼ Important

- Variabel biasa bisa diambil nilai pointernya, caranya dengan menambahkan tanda **ampersand** (&) tepat sebelum nama variabel. Metode ini disebut dengan **referencing**.
- Dan sebaliknya, nilai asli variabel pointer juga bisa diambil, dengan cara menambahkan tanda **asterisk** (*) tepat sebelum nama variabel. Metode ini disebut dengan **dereferencing**.