MODUL PRAKTIKUM 2 - I/O, TIPE DATA & VARIABEL

ALGORITMA DAN PEMROGRAMAN 1

S1 INFORMATIKA

Telkom University

Fakultas Informatika
School of Computing
Telkom University

informatics lab

LEMBAR PENGESAHAN

Saya yang bertanda tangan di bawah ini:

Nama	: Prasti Eko Yunanto, S.T., M.Kom.
NIP	: 19890017
Koordinator Mata Kuliah	: Algoritma dan Pemrograman 1
Prodi	: S1 Informatika

Menerangkan dengan sesungguhnya bahwa modul ini digunakan untuk pelaksanaan praktikum di Semester Ganjil Tahun Ajaran 2024/2025 di Laboratorium Informatika, Fakultas Informatika, Universitas Telkom.

Bandung, 17 Agustus 2024

Mengesahkan,	Mengetahui,
Koordinator Mata Kuliah	Kaprodi S1 Informatika
Algoritma Pemrograman 1	
Prasti Eko Yunanto, S.T., M.Kom.	Dr. Erwin Budi Setiawan, S.Si., M.T.
NIP. 19890017	NIP. 00760045

MODUL 2. I/O, TIPE DATA & VARIABEL

2.1 Input dan Output (I/O)

Salah satu Instruksi dasar di dalam pemrograman adalah input/scan/read dan output/print/write.

- **Input**

  Merupakan instruksi dasar untuk membaca data yang diberikan dari pengguna. Data yang diberikan oleh pengguna akan disimpan ke dalam suatu wadah yang disebut variabel. Data ini untuk selanjutnya akan diproses oleh program komputer.

  Pembacaan data ini tentunya memerlukan perangkat/device tambahan yang terhubung oleh komputer. Misalnya: keyboard, mouse, layar sentuh, dan lainnya. Informasi yang diketik oleh pengguna akan disimpan sebagai data komputer di dalam variabel.

  Penulisan instruksi input beragam menyesuaikan bahasa pemrograman yang digunakan (misalnya scan atau read). Penulisan di dalam bahasa pemrograman Go dapat dilakukan seperti contoh berikut ini:

  ```
  instruksi_input(variabel1, variabel2, variabel3, ...)
  ```


Notasi instruksi dasar	Notasi dalam bahasa Go
input(var_1)	**fmt.Scan**(&var_1)
	fmt.Scanf("%d", &var_1)
input(var_1, nilai_y)	**fmt.Scan**(&var_1, &nilai_y)
Keterangan: • var_1 dan nilai_y merupakan variabel atau wadah yang digunakan untuk menampung data. • Scanf merupakan instruksi input dengan format data tertentu, %d merupakan format untuk data berupa bilangan bulat. • Instruksi Scan lebih direkomendasikan apabila belum memahami string format.	

- **Output**

  Berbeda dengan Input, maka Output merupakan perintah untuk menampilkan data ke layar monitor* (bisa dari perangkat komputer ataupun mobile). Data yang sudah diproses atau diolah oleh program komputer perlu ditampilkan ke layar sehingga pengguna bisa memperoleh informasi dari hasil pengolahan data yang dilakukan oleh program.

Kata kunci instruksi yang digunakan juga beragam, menyesuaikan dengan bahasa pemrograman yang digunakan. Contohnya write dan print.

Catatan*: *Pada konteks ini, perangkat untuk menampilkan data dipersempit menjadi layar komputer, tujuannya untuk membuat mahasiswa lebih mudah memahapi instruksi output.*

instruksi_output(variabel1/data1, variabel2/data2, ...)

Notasi instruksi dasar	Notasi dalam bahasa Go	Tampilan
output("Praktikum",2024)	**fmt.Print**("Praktikum",2024)	Praktikum2024
	fmt.Println("Praktikum",2024)	Praktikum 2024
	fmt.Printf("Praktikum %v",2024)	Praktikum 2024
print("Praktikum") **print**("Algoritma")	**fmt.Println**("Praktikum") **fmt.Println**("Algoritma")	Praktikum Algoritma
	fmt.Print("Praktikum") **fmt.Print**("Algoritma")	PraktikumAlgoritma
Keterangan: • "Praktikum" dan "Algoritma" merupakan data teks. • 2024 merupakan data bilangan bulat. • Instruksi Println menambahkan spasi antar data yang ditampilkan, dan pointer berada di baris baru setelah menampilkan data. Sehingga seolah-olah menambahkan "Enter" atau baris baru diakhir instruksi. • Printf merupakan perintah output dengan format data tertentu.		

Contoh sebuah program dalam bahasa pemrograman Go (nama file hello2.go).

```
1  package main
2  import "fmt"
3  func main() {
4      var mk string = "Algoritma dan Pemrograman"
5      var kode, sks int
6      fmt.Print("Tuliskan kode MK dan SKS: ")
7      fmt.Scan(&kode, &sks)
8      fmt.Println("Kredit MK",kode,"-",mk,"1 adalah",sks,"SKS")
9  }
10
```

```
C:\users\go\src\hello>go build hello2.go
C:\users\go\src\hello>dir
Directory of C:\users\go\src\hello
 6/29/2019   7:15 PM         1,727  hello2.go
 6/29/2019   7:18 PM     2,198,528  hello2.exe
C:\users\go\src\hello>.\hello
Tuliskan kode MK dan SKS: CAK1BAB3 3
Kredit MK CAK1BAB3 – Algoritma dan Pemrograman 1 adalah 3 SKS
C:\users\go\src\hello>
```

2.2 Data, Variabel, dan Instruksi Dasar

Variabel adalah nama dari suatu lokasi di memori, yang data dengan tipe tertentu dapat disimpan.

- Nama variabel dimulai dengan huruf dan dapat dikuti dengan sejumlah huruf, angka, atau garisbawah.

  Contoh: **ketemu**, **found**, **rerata**, **mhs1**, **data_2**, **...**

Notasi tipe dasar	Tipe dalam Go	Keterangan
integer	int int8 int32 //rune int64 uint uint8 //byte uint32 uint64	bergantung platform 8 bit: -128..127 32 bit: -10^9..10^9 64 bit: -10^19..10^19 bergantung platform 0..255 0..4294967295 0..(2^64-1)
real	float32 float64	32bit: -3.4E+38 .. 3.4E+38 64bit: -1.7E+308 .. 1.7E+308
boolean (atau logikal)	**bool**	**false** dan **true**
karakter	byte //uint8 rune //int32	tabel ASCII/UTF-8 tabel UTF-16
string	string	

- Tipe data yang umum tersedia adalah integer, real, boolean, karakter, dan string. Lihat tabel berikut ini untuk variasi tipe data yang disediakan dalam bahasa Go.
- **Nilai data** yang tersimpan dalam variabel dapat diperoleh dengan menyebutkan langsung nama variabelnya.

  Contoh: Menyebutkan nama **found** akan mengambil nilai tersimpan dalam memori untuk variabel **found**, pastinya.

- **Informasi alamat** atau lokasi dari variabel dapat diperoleh dengan menambahkan prefiks **&** di depan nama variabel tersebut.

  Contoh: **&found** akan mendapatkan alamat memori untuk menyimpan data pada **found**.

- Jika variabel berisi alamat memori, prefiks ***** pada variabel tersebut akan memberikan nilai yang tersimpan dalam memori yang lokasinya disimpan dalam variabel tersebut.

Contoh: ***mem** akan mendapatkan data di memori yang alamatnya tersimpan di **mem.** Karenanya ***(&found)** akan mendapatkan data dari lokasi memori variabel **found** berada, alias sama saja dengan menyebutkan langsung **found** 8=).

- Operasi yang dapat dilakukan terhadap tipe data di atas adalah:

Operator dalam Go	Tipe data terkait	Keterangan
+	string integer dan real	konkatenasi 2 string operasi penjumlahan
- * /	integer dan real	operasi pengurangan, perkalian, dan pembagian
%	integer	operasi sisa pembagian integer (modulo)
& | ^ &^	integer	operasi **per-bit** AND, OR, XOR, AND-NOT
<< >>	integer dan unsigned integer	operasi geser bit kiri/kanan sebanyak unsigned integer yang diberikan
< <= >= > == !=	selain boolean	komparasi menghasilkan nilai boolean komparasi karakter sesuai dengan posisi karakter tersebut dalam tabel ASCII/UTF-16 komparasi string sesuai dengan operasi karakter per karakter, dimulai dari karakter paling kiri (awal)
&& || !	boolean	operasi **boolean** AND, OR, dan NOT
* &	variabel apasaja	mendapatkan data dari lokasi memori dan mendapatkan lokasi dari variabel

Contoh:

Operasi	Hasil
"non suffi" + "cit mundo"	"non sufficit mundo"
2019.01 + 1.0102	2020.0202
2020 / 20	22.22
20.2 * 1.1	101
2020 % 1999	21
2020 & 1111	2104
2020 ^ 1111	1663
2020 >> 2	505
"minutus" < "magnus"	false
2020 >= 1234	true
! false && true	true

- Bahasa Go menganut kesesuaian tipe data yang ketat. Tipe data yang berbeda tidak boleh dicampur dalam satu ekspresi, bahkan tipe data masih yang sejenis, misalnya masih sama-sama integer (**int** dan **int32**). Untuk menyesuaikan tipe data, ada beberapa cara yang dapat dilakukan:
  - ✓ Casting, **tipe(data),** mengubah tipe dari data yang diberikan ke tipe yang diinginkan.
  - ✓ Memanfaatkan fungsi **Sprint** dan **Sscan** dari paket **fmt**.
  - ✓ Memanfaatkan fungsi-fungsi dalam paket **strconv**, seperti **Atoi**, **Itoa**, dan **ParseBool**. Lihat lampiran untuk contoh penggunaan.

  Contoh:

Operasi	Hasil
2020.0 % 19	will be an illegal expression error
int(2020.0) % 19	6

Konversi tipe	Data	Tipe baru	Keterangan
tipe(data)	integer	integer	format data tidak berubah, hanya penyesuaian jumlah bit. Kekurangan bit diisi bit 0 di sebelah kiri (MSB)
	real	real	format data tidak berubah, hanya penyesuaian jumlah bit. Kekurangan bit, maka bit mantisa diisi bit 0.
	real	integer	format data disesuaikan dengan tipe data tujuan
	integer	real	format data disesuaikan dengan tipe data tujuan
fmt.Sprintf("%v",v)	any type	string	tulis output ke string
fmt.Sprintf("%c",v)	karakter	string	tulis karakter ke string
fmt.Sscanf(s,"%v",&v)	string	any type	baca string ke variabel dengan tipe tertentu
fmt.Sscanf(s,"%c",&v)	string	karakter	baca string ke variabel bertipe karakter

- Variabel harus dideklarasikan dulu sebelum digunakan. Variabel juga harus diinisialisasi dulu (diisi data) agar nilai yang tersimpan diketahui dengan jelas dan eksekusi algoritma menjadi terprediksi. Dalam bahasa Go, variabel yang tidak diinisialisasi lebih dahulu otomatis diisi dengan nilai default yang ekuivalen dengan bit 0.

- ✓ Nilai 0 untuk bilangan integer
- ✓ 0.0E+0 untuk bilangan real
- ✓ **false** untuk boolean
- ✓ Karakter NUL (lihat tabel ASCII) untuk karakter
- ✓ "" (string kosong, string dengan panjang 0) untuk string
- ✓ **nil** untuk alamat memori

- Variabel bisa dianalogikan sebagai wadah yang akan digunakan untuk menyimpan data.

Notasi deklarasi variabel	Penulisan dalam Go	Keterangan
kamus a : tipe	**var** a tipe	a diinisialisasi dengan nilai default

kamus
  a : tipe
 	
algoritma
  a = nilai_awal
	var a tipe = nilai_awal
	var a = (tipe)nilai_awal
		a diinisialisasi dengan nilai_awal

	a := nilai_awal a := (tipe)nilai_awal	secara **implisit**, tipe variabel a ditentukan dari nilai inisialisasinya

Contoh:

```
1  func main() {
2      var a int
3      a = 2019
4      r := 2019.0707
5      b := false
6      c := 'x'
7      s := "a string is a string"
8  }
```

- Instruksi Dasar

Notasi instruksi dasar	Penulisan dalam bahasa Go	Keterangan
v1 = e1 v1 = v1 + e1 v1 = v1 – e1 v1 = v1 + 1 v1 = v1 - 1	v1 = e1 v1 += e1 // atau v1 = v1 + e1 v1 -= e1 // atau v1 = v1 – e1 v1++ // atau v1 = v1 + 1 v1-- // atau v1 = v1 - 1	operasi assignment, mengisi data ke lokasi memori (variabel)

input(v1, v2)	fmt.**Scan**(&v1, &v2) fmt.**Scanln**(&v1, &v2) fmt.**Scanf**(“%v %v”, &v1, &v2)	Pembacaan data memerlukan alamat memori ke mana data akan disimpan.
output(e1, e2)	fmt.**Print**(e1, e2) fmt.**Println**(e1, e2) fmt.**Printf**(“%v %v\n”, e1, e2)	Penulisan data memerlukan nilai data yang akan ditulis.

Contoh:

```
1   package main
2   import “fmt”
3
4   func main() {
5       var a, b, c float64
6       var hipotenusa bool
7
8       fmt.Scanln( &a, &b, &c )
9       hipotenusa = (c*c) == (a*a + b*b)
10      fmt.Println( “Sisi c adalah hipotenusa segitiga a,b,c: “, hipotenusa)
11  }
```

2.3 Konstanta Simbolik

Konstanta dapat diberi nama untuk memudahkan mengingat maksud dan manfaat dari nilai yang diberi nama tersebut. Seperti PI untuk merepresentasikan konstanta π.

```
1  const PI = 3.14159265358979323846264433
2  const MARKER = “AKHIR”
```

2.4 Tipe Data Character

Tipe data character pada dasarnya merupakan tipe data integer, tetapi digunakan untuk merepresentasikan suatu simbol tertentu berdasarkan tabel acuan ASCII/UTF-8 atau UTF-16.

Notasi tipe dasar	Tipe dalam Go	Keterangan
char	byte //uint8 rune //int32	tabel ASCII/UTF-8 tabel UTF-16

Tabel 1. Contoh potongan tabel ASCII untuk beberapa bilangan integer

	0	1	2	3	4	5	6	7
32	SPC	!	“	#	$	%	&	'
40	(	)	*	+	,	-	.	/
48	0	1	2	3	4	5	6	7
56	8	9	:	;	<	=	>	?

64 **72**	@ H	A I	B J	C K	D L	E M	F N	G O
80 **88**	P X	Q Y	R Z	S [	T \	U]	V ^	W _
96 **104**	` h	a i	b j	c k	d l	e m	f n	g o
112 **120**	p x	q y	r z	s {	t |	u }	v ~	w DEL

Simbol pada tipe data character ditulis dengan menambahkan petik satu, Contohnya bilangan 64 dapat ditulis dengan simbol '@' (dengan petik) dan bilangan 97 dapat ditulis dengan simbol 'a'.

2.5 Contoh Soal Modul 2

Berikut adalah beberapa contoh soal terkait penggunaan tipe data integer, character dan operasinya.

1) Buatlah program yang digunakan untuk menghitung hasil penjumlahan 5 bilangan bulat.

   Masukan terdiri dari lima bilangan bulat a, b, c, d, dan e.

   Keluaran berupa bilangan hasil penjumlahan lima bilangan bulat a, b, c, d, dan e.

   Contoh masukan dan keluaran:

No.	Masukan	Keluaran
1	3 2 7 10 2	24
2	11 22 33 44 55	165

   Jawaban:

```
1   // filename : penjumlahan.go
2   package main
3   import “fmt”
4
5   func main() {
6       var a, b, c, d, e int
7       var hasil int
8       fmt.Scanln( &a, &b, &c, &d, &e)
9       hasil = a + b + c + d + e
10      fmt.Println("Hasil penjumlahan",a,b,c,d,e,"adalah",hasil)
11  }
C:\users\go\src\hello>go build penjumlahan.go
C:\users\go\src\hello>.\penjumlahan
11 22 33 44 55
Hasil penjumlahan 11 22 33 44 55 adalah 165
```

2) Sebuah program digunakan untuk menghitung persamaan $f(x) = \frac{2}{x+5} + 5$.

Masukan terdiri dari sebuah bilangan bulat.

Keluaran berupa bilangan yang menyatakan nilai dari $f(x)$.

Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	5	5.2
2	-23	4.888888888888888889

Jawaban:

```
package main
import "fmt"

func main(){
    var x, fx float64
    fmt.Scan(&x)
    fx = 2/(x+5) + 5
    fmt.Println(fx)
}
```


3) Tipe karakter sebenarnya hanya apa yang tampak dalam tampilan. Di dalamnya tersimpan dalam bentuk biner 8 bit (byte) atau 32 bit (rune) saja.
Buat program ASCII yang akan membaca 5 buat data integer dan mencetaknya dalam format karakter. Kemudian membaca 3 buah data karakter dan mencetak 3 buah karakter setelah karakter tersebut (menurut tabel ASCII)

Masukan terdiri dari dua baris. Baris pertama berisi 5 buah data integer. Data integer mempunyai nilai antara 32 s.d. 127. Baris kedua berisi 3 buah karakter yang berdampingan satu dengan yang lain (tanpa dipisahkan spasi).

Keluaran juga terdiri dari dua baris. Baris pertama berisi 5 buah representasi karakter dari data yang diberikan, yang berdampingan satu dengan lain, tanpa dipisahkan spasi. Baris kedua berisi 3 buah karakter (juga tidak dipisahkan oleh spasi).

Contoh masukan dan keluaran:

No.	Masukan	Keluaran
1	66 97 103 117 115 SNO	Bagus TOP

Catatan: Gunakan `fmt.Scanf("%c", &var)` untuk pembacaan satu karakter dan `fmt.Printf("%c", var)` untuk penulisan satu karakter.

Jawaban:

```
1   // filename : ascii.go
2   package main
3   import "fmt"
4   func main(){
5       var c1, c2, c3, c4, c5 byte
6       var b1, b2, b3 int
7
8       fmt.Scan(&c1, &c2, &c3, &c4, &c5)
9       fmt.Scanf("%c",&b1)
10      fmt.Scanf("%c",&b2)
11      fmt.Scanf("%cc",&b3)
12
13      fmt.Printf("%c%c%c%c%c",c1, c2, c3, c4, c5)
14      fmt.Printf("%c%c%c",b1+1, b2+1, b3+1)
15  }
16
```

```
C:\users\go\src\hello>go
build ascii.go
C:\users\go\src\hello>.\
ascii
11 22 33 44 55
66 97 103 117 115
SNO
Bagus
TOP
C:\users\go\src\hello>
```