

*Berdo'alah sebelum mengerjakan. Dilarang berbuat curang.  
Tugas ini untuk mengukur kemampuan anda, jadi kerjakan dengan sepenuh hati.  
Selamat belajar, semoga sukses !*

<b>Nama Mahasiswa:</b> <b>Meilyand Evriyan Timor</b> .....	<b>NIM:</b> <b>1301161769</b> .....	<b>Nilai:</b> .....
<b>Nama Mahasiswa:</b> <b>Reyhan Rahmansyah</b> .....	<b>NIM:</b> <b>1301160805</b> .....	<b>Nilai:</b> .....
<b>Nama Mahasiswa:</b> <b>Reno butar Butar</b> .....	<b>NIM:</b> <b>1301164724</b> .....	<b>Nilai:</b> .....

**Siapkan tools berikut sebelum mengerjakan:**

1. Go Programming Language (<https://golang.org/dl/>).
2. Visual Studio Code (<https://code.visualstudio.com/>) atau LiteIDE (<https://github.com/visualfc/liteide>).
3. Harus menggunakan linux dengan distro fedora (<https://getfedora.org/id/workstation/>).
4. Buatlah git repository pada <https://github.com/> kemudian push semua kode dan hasil laporan anda ke dalam repository github yang sudah anda buat.
5. Lakukan instalasi flatbuffer (<https://google.github.io/flatbuffers/>) untuk mengerjakan salah satu tugas pada modul ini.
6. Kumpulkan link repository github tersebut sebagai tanda bahwa anda mengerjakan tugas modul ini.
7. Link repository harus berbeda untuk setiap tugasnya. Buatlah markdown yang rapi di setiap repository tugas yang anda kumpulkan.
8. Printscreen program harus dari desktop kelompok anda sendiri, dan harus dari linux yang sudah diinstall. Jika tidak, maka harus mengulang pengerjaan tugasnya.
9. Jangan lupa untuk menuliskan NAMA dan NIM pada laporan.
10. Laporan berbentuk PDF dan dikumpulkan pada link repository github beserta kodenya.
11. Walaupun tugas berkelompok tapi pengumpulan link github harus individu, jika tidak mengumpulkan maka dianggap tidak mengerjakan.

Nama:	NIM:	Nilai:
-------	------	--------

### Soal No 1 (JSON Marshal)

```
package main

import (
    "encoding/json"
    "fmt"
)

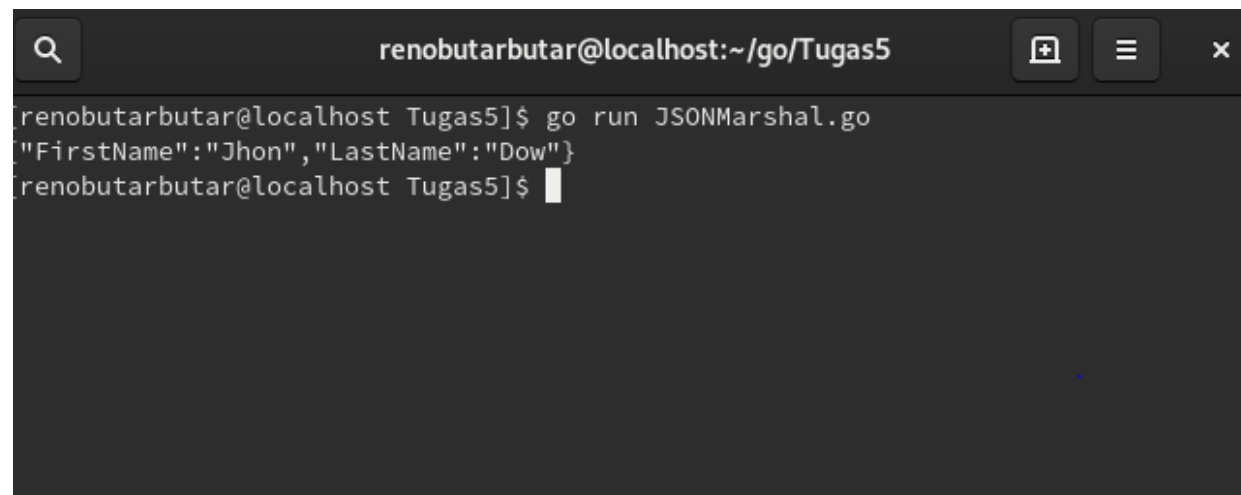
type Person struct {
    FirstName string `json:"firstName"`
    LastName  string `json:"lastName"`
}

func main() {
    bytes, err := json.Marshal(Person{
        FirstName: "John",
        LastName:  "Dow",
    })
    if err != nil {
        panic(err)
    }

    fmt.Println(string(bytes))
}
```

Jalankan program diatas, apakah outputnya (berikan printscreen) dan jelaskan cara kerjanya!

Jawaban:



```
renobutarbutar@localhost:~/go/Tugas5
renobutarbutar@localhost Tugas5]$ go run JSONMarshal.go
{"FirstName": "Jhon", "LastName": "Dow"}
renobutarbutar@localhost Tugas5]$
```

Cara kerjanya :

Fungsi `json.Marshal` digunakan untuk decoding data ke json string. Sumber data bisa berupa variabel objek cetakan struct, `map[string]interface{}`, atau slice.

Kemudian Struct `Person` diinisialisasi pada variabel `bytes` dengan atribut nama depan (`FirstName`) "John" dan nama belakang (`LastName`) "Dow". Kemudian, `bytes` diserialisasikan kedalam bentuk JSON dan dicetak menghasilkan keluaran struct `Person` dalam bentuk JSON `{{"FirstName": "John", "LastName": "Dow"}}`.

Nama:	NIM:	Nilai:
-------	------	--------

--

Nama:	NIM:	Nilai:
-------	------	--------

### Soal No 2 (JSON Unmarshal)

```
package main

import (
    "encoding/json"
    "fmt"
)

type Person struct {
    FirstName string `json:"firstName"`
    LastName  string `json:"lastName"`
}

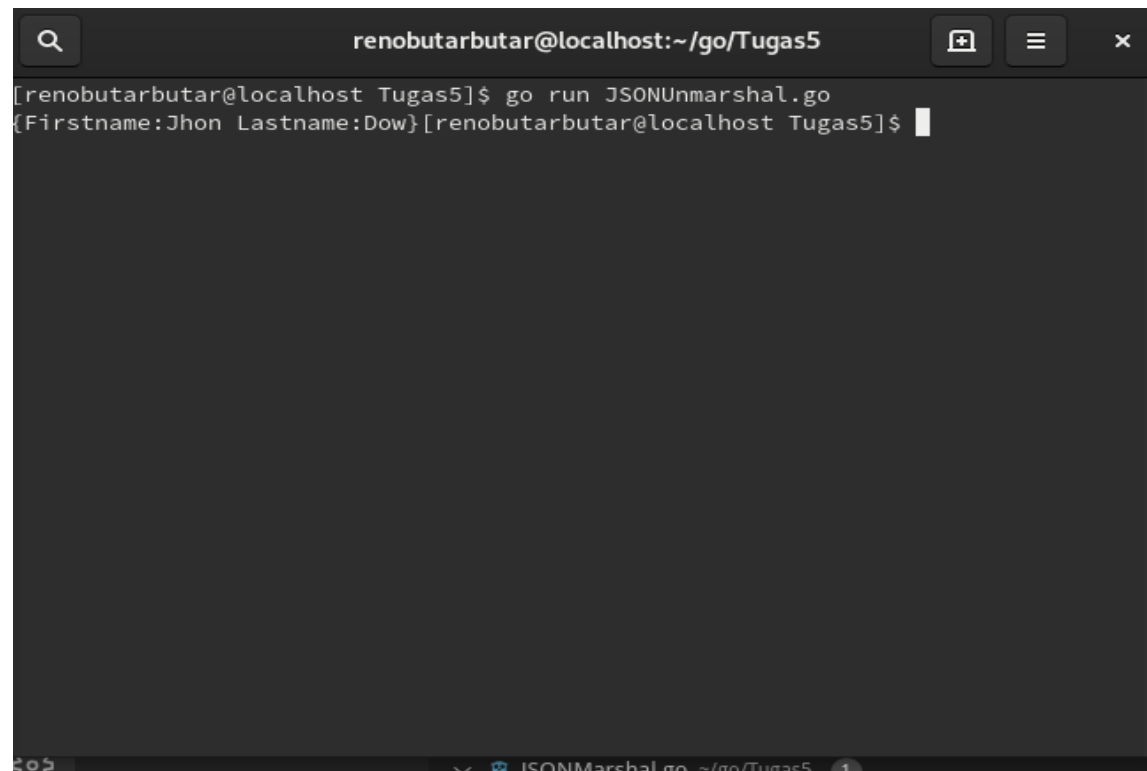
func main() {
    in := `{"firstName":"John","lastName":"Dow"}`
    bytes := []byte(in)

    var p Person
    err := json.Unmarshal(bytes, &p)
    if err != nil {
        panic(err)
    }

    fmt.Printf("%+v", p)
}
```

Jalankan program diatas, apakah outputnya (berikan printscreen) dan jelaskan cara kerjanya!

Jawaban:



```
renobutarbutar@localhost:~/go/Tugas5
[renobutarbutar@localhost Tugas5]$ go run JSONUnmarshal.go
{Firstname:Jhon Lastname:Dow}[renobutarbutar@localhost Tugas5]$
```

Nama:	NIM:	Nilai:
-------	------	--------

Cara Kerja :

Data json dituliskan sebagai string. Dengan menggunakan json.Unmarshal, json string bisa dikonversi menjadi bentuk objek. Struct Person dalam bentuk JSON di-assign ke variabel in, yang kemudian variabel in menjadi nilai untuk variabel bytes dalam tipe data []byte. Kemudian, variabel bytes di decode dari bentuk JSON menjadi bentuk struct Person dan ditampung pada variabel p. Sehingga, keluaran yang dihasilkan dari mencetak variabel p adalah {Firstname:John Lastname:Dow}.

### Soal No 3 (Flatbuffer dan Protocol Buffer)

Jalankan program pada repository github berikut: <https://github.com/jonog/grpc-flatbuffers-example>

Berikan analisis berupa:

1. Apakah outputnya (berikan printscreen)!
2. Jelaskan cara kerjanya dan buatlah diagram FSMnya!
3. Analisis perbedaan dari protocol buffer dan flatbuffer!

Nama:	NIM:	Nilai:
-------	------	--------

Jawaban:

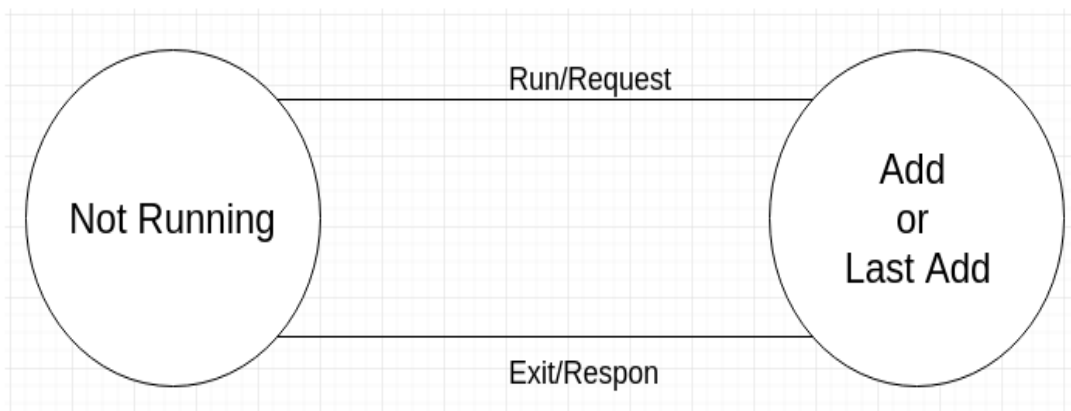
1. Hasil

```

renobutarbutar@localhost:~/go/grpc-flatbuffers-example-master
[renobutarbutar@localhost 5_3]$ cd $HOME
[renobutarbutar@localhost ~]$ cd go
[renobutarbutar@localhost go]$ ls
5_3  bin  grpc-flatbuffers-example-master  one.go  pkg  src  Tugas5
[renobutarbutar@localhost go]$ cd grps-flatbuffers-example-master
bash: cd: grps-flatbuffers-example-master: No such file or directory
[renobutarbutar@localhost go]$ cd grpc-flatbuffers-example-master
[renobutarbutar@localhost grpc-flatbuffers-example-master]$ make compile
cd bookmarks-client && go build -o ../client && cd ..
client.go:8:2: cannot find package "github.com/google/flatbuffers/go" in any of:
    /usr/lib/golang/src/github.com/google/flatbuffers/go (from $GOROOT)
    /home/renobutarbutar/go/src/github.com/google/flatbuffers/go (from $GOPATH)
client.go:9:2: cannot find package "github.com/jonog/fbs-example/bookmarks" in any of:
    /usr/lib/golang/src/github.com/jonog/fbs-example/bookmarks (from $GOROOT)
    /home/renobutarbutar/go/src/github.com/jonog/fbs-example/bookmarks (from $GOPATH)
client.go:11:2: cannot find package "google.golang.org/grpc" in any of:
    /usr/lib/golang/src/google.golang.org/grpc (from $GOROOT)
    /home/renobutarbutar/go/src/google.golang.org/grpc (from $GOPATH)
make: *** [Makefile:12: compile_bookmarks_client] Error 1
[renobutarbutar@localhost grpc-flatbuffers-example-master]$

```

2. FSM



3. Perbedaan dari Protocol buffer dan flatbuffer

Flatbuffer Perbedaan utama antara protobuf dan buffer datar adalah bahwa tidak perlu melakukan deserialisasi seluruh data pada yang terakhir sebelum mengakses objek. Ini membuat buffer rata

Nama:	NIM:	Nilai:
-------	------	--------

bertanggung jawab atas kasus penggunaan yang memiliki banyak data. Ini juga mengkonsumsi memori jauh lebih sedikit daripada protobuf. Kode juga merupakan urutan besarnya lebih kecil dan mendukung lebih banyak fitur skema (mis. Serikat jenis dalam XML) Flatbuffers juga menderita kelemahan yang sama dengan protobuf karena kurangnya representasi yang dapat dibaca manusia

Nama:	NIM:	Nilai:
-------	------	--------

--