Laboratorio 10 - Strutture

1 Qual è l'output?

Qual è l'output del seguente programma?

```go
package main

import (
    "fmt"
)

type Persona struct {
    Nome    string
    Cognome string
}

func main() {

    var p1 Persona
    p1 = Persona{"Rick", "Sanchez"}

    var p2 Persona
    p2.Nome = "Jerry"
    p2.Cognome = "Smith"

    p3 := Persona{Nome: "Morty"}

    p2 = p3

    fmt.Println(p1, p2)
}
```

2 Qual è l'output?

Qual è l'output del seguente programma?

```go
package main

import (
    "fmt"
)

type Persona struct {
    Nome    string
    Cognome string
}

func main() {

    anagrafica := []Persona{
        {"Rick", "Sanchez"},
        {"Morty", "Smith"},
        {"Jerry", "Smith"},
        {"Summer", "Smith"},
        {"Beth", "Smith"}}

    CambiaPrimoCognome(anagrafica)

    CambiaCognome(anagrafica[0])

    fmt.Println(anagrafica[0])
```

```
}

func CambiaPrimoCognome(a []Persona) {
    a[0].Cognome = "Martinez"
}

func CambiaCognome(p Persona) {
    p.Cognome = "Gomez"
}
```

3 Rubrica (1)

Si consideri una rubrica in cui:

- Ogni contatto deve avere un cognome, un nome, un indirizzo ed un numero di telefono.
- Ogni indirizzo deve contenere le seguenti informazioni: via, numero civico, CAP e città.
- Non possono esistere due contatti con lo stesso cognome e lo stesso nome.

Strutture dati da definire

Definire i seguenti tipi di dati:

- `Indirizzo`: una struttura che memorizzi un indirizzo nei seguenti campi: `via`, `cap` e `città` di tipo `string` e `numeroCivico` di tipo `uint`;
- `Contatto`: una struttura che memorizzi i dati di un contatto nei seguenti campi: `cognome`, `nome` e `telefono` di tipo `string` e `indirizzo` di tipo `Indirizzo`;
- `Rubrica`: una slice dove ogni elemento è di tipo `Contatto`. Ogni elemento della slice rappresenta un contatto inserito nella rubrica. Una slice vuota rappresenta una rubrica vuota.

Funzioni da implementare

Implementare le funzioni:

- `NuovaRubrica() Rubrica` che restituisce un valore `Rubrica` rappresentante una rubrica senza alcun contatto inserito;
- `InserisciContatto(r Rubrica, cognome, nome string, via string, numero uint, cap, città string, telefono string) Rubrica` che data la rubrica `r` restituisce una nuova rubrica in cui è inserito il nuovo contatto creato con i dati passati come argomento. Se la rubrica `r` contiene già un contatto con identici `nome` e `cognome` non avviene nessuna modifica e la rubrica restituita è `r` stessa;
- `EliminaContatto(r Rubrica, cognome, nome string) Rubrica` che restituisce una rubrica in cui è eliminato il contatto avente `nome` e `cognome` uguali a quelli passati in input. Se tale contatto non esiste, la rubrica restituita è `r` stessa;
- `StampaContatto(c Contatto)` che stampa a video i dettagli del contatto `c` nel seguente formato: `nome cognome: via numeroCivico, città, cap - Tel. telefono\n`;
- `StampaRubrica(r Rubrica)` che stampa a video tutti i contatti presenti nella rubrica utilizzando per ogni contatto la funzione `StampaContatto()`. La stampa deve rispettare il seguente formato:

  ```
  Rubrica:\n
  [1] - nome cognome: via numeroCivico, città, cap - Tel. telefono\n
  [2] - nome cognome: via numeroCivico, città, cap - Tel. telefono\n
  [3] - nome cognome: via numeroCivico, città, cap - Tel. telefono\n
  ```

- `AggiornaContatto(rubrica Rubrica, cognome, nome string, via string, numero uint, cap, città string, telefono string) Rubrica` che aggiorna i dettagli del contatto identificato da `nome`

e `cognome` e restituisce la rubrica con il contatto aggiornato. Se il contatto non esiste, viene restituita la rubrica `r` stessa.

Esempio d'esecuzione:

Il formato dell'output dopo alcune operazioni di inserimento, cancellazione e modifica della rubrica dovrebbe risultare simile al seguente:

```
$ go run rubrica.go
Rubrica:
[1] - Mario Rossi: Via Festa del Perdono 11, 20122, Milano - Tel. 02503111
[2] - Anna Rossi: Via Festa del Perdono 11, 20122, Milano - Tel. 02503111
[3] - Carlo Rossi: Via Festa del Perdono 11, 20122, Milano - Tel. 02503111
Rubrica:
[1] - Anna Rossi: Via Festa del Perdono 11, 20122, Milano - Tel. 02503111
[2] - Carlo Rossi: Via Festa del Perdono 11, 20122, Milano - Tel. 02503111
Rubrica:
[1] - Anna Rossi: Via S. Sofia 25, Milano, 20122 - Tel.
[2] - Carlo Rossi: Via Festa del Perdono 11, 20122, Milano - Tel. 02503111
```

4 Rubrica (2)

Si consideri la rubrica dell'esercizio **3 Rubrica (1)**.

Si modifichi il programma in modo tale che le operazioni sulla rubrica vengano gestite tramite dei comandi letti da **standard input**.

Il programma deve:

- creare una rubrica vuota;
- leggere da **standard input** un testo su più righe (ogni riga è un comando);
- terminare la lettura quando viene inserito da standard input l'indicatore EOF (CTRL+D);
- eseguire in sequenza i comando letti eseguendo le funzioni corrispondenti.

Ogni comando è una riga di testo in cui il primo carattere determina il tipo dell'operazione:

- `I` : inserimento di un contatto
- `E` : cancellazione di un contatto
- `S` : stampa della rubrica
- `A` : aggiornamento di un contatto

I valori successivi rappresentano i valori da assegnare agli argomenti delle funzioni che effettuano l'operazione richiesta (le funzioni implementate nell'esercizio **3 Rubrica (1)**. I valori sono sempre separati tra di loro dal carattere `;` .

Inserimento

Il comando per l'inserimento di un nuovo contatto ha il seguente formato:

```
I;cognome;nome;via;numeroCivico;cap;città;telefono
```

Cancellazione

Il comando per la cancellazione di un nuovo contatto ha il seguente formato:

```
E;cognome;nome
```

Stampa

Il comando per la stampa della rubrica è il seguente:

```
S
```

Aggiornamento di un contatto

```
A;cognome;nome;via;numeroCivico;cap;città;telefono
```

Suggerimento: per separare i valori presenti all'interno di una riga di testo utilizzate la funzione `strings.Split()` (usate `go doc strings.Split` per leggerne la documentazione).

Esempio d'esecuzione:

```
$ cat comandi.txt
I;Mario;Rossi;Via Celoria;18;20122;Milano;02503111
I;Mario;Rossi;Via Celoria;18;20122;Milano;
S
I;Elena;Bianchi;Via Celoria;18;20122;Milano;02503111
S
E;Mario;Rossi
S
A;Elena;Bianchi;Via Festa del perdono;7;20122;Milano;02503111
S

$ go run rubrica.go < comandi.txt
Rubrica:
[1] - Mario Rossi: Via Celoria 18, 20122, Milano - Tel. 02503111
Rubrica:
[1] - Mario Rossi: Via Celoria 18, 20122, Milano - Tel. 02503111
[2] - Elena Bianchi: Via Celoria 18, 20122, Milano - Tel. 02503111
Rubrica:
[1] - Elena Bianchi: Via Celoria 18, 20122, Milano - Tel. 02503111
Rubrica:
[1] - Elena Bianchi: Via Festa del perdono 7, Milano, 20122 - Tel. 02503111
```

5 Rubrica (3)

Si consideri l'esercizio **4 Rubrica (2)**. Modificare il programma in modo tale che il tipo `Rubrica` sia implementato come una `map[string]Contatto` invece che come una slice `[]Contatto`.

Nella nuova implementazione la chiave di tipo `string` della mappa sarà la stringa ottenuta concatenando `nome` e `cognome` del `Contatto` associato.