Lab04 – Iterazione/Stringhe – II parte

Esercizio 1 - Carte

Sapendo che al codice Unicode 127163 (associato alla rappresentazione in bit Unicode/UTF-8 '\U0001F0B1') corrisponde il simbolo "asso di cuori", e che i codici successivi corrispondono alle carte successive (2 di cuori, 3 di cuori, …), scrivere il codice che stampa tutte le carte da gioco dall'asso di cuori al 10 di cuori (e, se gli argomenti sono stati trattati a lezione, l'equivalente codice Unicode in base 10 e 16, nonché l'equivalente rappresentazione in bit Unicode/UTF-8).

Suggerimento: un carattere è una variabile di tipo `rune`, il cui valore è un intero corrispondente al codice Unicode del carattere. Le istruzioni equivalenti `var c rune = 127163` e `var c rune = '\U0001F0B1'` servono a definire la varibile `c` di tipo `rune` ed inizializzarla al valore "asse di cuori". Per stampare la carta da gioco "asse di cuori" si può utilizzare l'instruzione `fmt.Print(string(c))`.

Esercizio 2 – Saluto con data

Create un programma, che chiede all'utente di inserire il suo nome (che chiameremo `NOME`) e gli risponde con:

Ciao `NOME`, sai che oggi è `DATA`?

Dove `NOME` è il nome inserito dall'utente, e `DATA` è la data di oggi. Per generare la data di oggi usate la funzione `time.Now()` del package `time`.

Esempio di esecuzione (in grassetto il nome inserito dall'utente)***:***

```
>go run Soluzione
Come ti chiami?
Marco
Ciao marco, sai che oggi è il 2018-05-16 06:58:13.0470322 +0200 CEST m=+3.244683601
```

Esercizio 3 – Trasformazione stringhe in minuscole/maiuscole

Scrivete un programma che legga una stringa inserita dall'utente e, considerando l'insieme C di tutti caratteri che corrispondono a lettere dell'alfabeto inglese, riscriva la stringa:

- con caratteri tutti maiuscoli se i caratteri appartengono a C;
  ad esempio, la stringa:
  "`MA... che !!! bello !!! andar al MAR!`" diventa:
  "`MA... CHE !!! BELLO !!! ANDAR AL MAR!`"

- con caratteri tutti minuscoli se i caratteri appartengono a C;
  ad esempio, la stringa:
  “MA... che !!! bello !!! andar al MAR!” diventa:
  “ma... che !!! bello !!! andar al mar!”

SUGGERIMENTI:

1) Per leggere una qualsiasi riga di testo, i.e., una stringa in cui compaiono caratteri spazio e terminata da ‘\n’,si può utilizzare il seguente programma:

```
import (
    "bufio"
    "fmt"
    "os"
)

func main() {

  fmt.Println("Introduci la riga di testo:")

  str := ""
  scanner := bufio.NewScanner(os.Stdin)
  if scanner.Scan() {

        str = scanner.Text();
        str += "\n"

  }
  fmt.Println("Riga di testo letta: ")
  fmt.Print(str)

}
```

2) **Notate che i caratteri che vengono modificati sono solo le lettere dell’alfabeto inglese...** Per riconoscere le lettere sfruttate il fatto che i caratteri sono rappresentati da variabili di tipo `rune`, il cui valore è un intero corrispondente al codice Unicode del carattere (gli interi da 0 a 127 corrispondono al codice dei caratteri considerati nello standard US-ASCII, integrato nello standard Unicode). In particolare, per le lettere dell’alfabeto inglese il codice Unicode (US-ASCII) è nei range:
   a. `[65,90]` per le lettere MAIUSCOLE (`‘A’=65` e `‘Z’=90`)
   b. `[97, 122]` per le lettere minuscole (`‘a’=97` e `‘z’=122`)

   Sia `c` una varibile di tipo `rune`, i seguenti blocchi di codici sono sintatticamente/semanticamente corretti:

```
if (c >='A' && c<='Z') || (c>='a' && c<='z'){
        fmt.Println(string(c), "è una lettera dell’alfabeto inglese!")
}else{
        fmt.Println(string(c), "non è una lettera dell’alfabeto inglese!")
```

```
}
// Si noti che la funzione unicode.IsLetter(c) del package unicode potrebbe restituire 'true'
// anche per caratteri che corrispondono a lettere ma che non appartenengono all'alfabeto
// inglese


if (c >='A' && c <= 'Z') {
        fmt.Println("L'equivalente carattere minuscolo è:", string('a' + (c - 'A')))
}
// ... oppure, utilizzando il package "unicode"...
if (c >='A' && c <= 'Z') {
        fmt.Println("L'equivalente carattere minuscolo è:", string(unicode.ToLower(c)))
}
```

3) Per accedere a tutti i caratteri che fanno parte di una stringa `str` si utilizza il costrutto:

```
for _,c := range str {
        // 'c' è una variabile di tipo rune che conterrà all'i-esima iterazione il valore
        // dell'i-esimo carattere nella string str
}
```

Esercizio 4 – Analisi di stringa

Scrivete un programma che legga una stringa inserita dall'utente e, considerando l'insieme C di tutti caratteri che corrispondono a lettere dell'alfabeto inglese, stampi

- il numero di lettere maiuscole e il numero di lettere minuscole nella stringa.
- il numero di consonanti e il numero di vocali della stringa.

Esercizio 5 – Analisi di stringa - Variante

Scrivete un programma che legga una stringa inserita dall'utente e, considerando l'insieme C di tutti caratteri che corrispondono a lettere dell'alfabeto inglese, stampi:

- il numero di VOCALI maiuscole e il numero di vocali minuscole nella stringa.
- il numero di CONSONANTI maiuscole e il numero di consonanti minuscole della stringa.

Esercizio 6 – Stringhe alternate

Scrivete un programma che legga una stringa inserita dall'utente e, considerando l'insieme C di tutti caratteri che corrispondono a lettere dell'alfabeto inglese, riscriva la stringa con ogni carattere in posizione pari in maiuscolo (se

il carattere appartiene a C) e ogni carattere in posizione dispari in minuscolo (se il carattere appartiene a C). Il primo carattere, in posizione 0, è da considerarsi in posizione pari.

Ad esempio, la stringa
"MA..._che_bello!!!_andar_al_MAR!" diventa:
"Ma..._ChE_BeLlO!!!_aNdAr_aL_MaR!"

Esercizio 7 – Sequenza inversa di numeri

Scrivere un programma che prenda in ingresso da standard input una sequenza di cifre intere strettamente positive (una cifra per linea). La sequenza $x_0, x_1, \ldots, x_n$ contiene cifre di valore compreso tra 1 e 9. L'inserimento termina dopo l'inserimento di uno 0. Il programma non considera eventuali numeri negativi o numeri positivi maggiori di 9. Mentre legge la sequenza, il programma calcola il valore:

$sum = x_0 - x_1 + x_2 - x_3 + x_4 - \ldots x_n$

ovvero la somma dei numeri in cui i numeri di indice dispari (il secondo, il quarto, il sesto,…) vengono moltiplicati per (-1).

Terminata la lettura il programma calcola anche la somma:

$sumInversa = x_n - x_{n-1} + x_{n-2} - x_{n-3} + x_{n-4} - \ldots x_0$

ovvero la somma dei valori in cui il penultimo, il terzultimo, etc… numeri sono moltiplicati per -1 (se l'utente inserisce un numero dispari di cifre le due somme sono in realtà uguali!).

Al termine dei calcoli, il programma stampa: la sequenza di cifre nell'ordine in cui sono state lette, la sequenza di cifre nell'ordine inverso e i risultati delle due somme.

SUGGERIMENTI

1) Sia `str` una varibile di tipo `string`, il seguenti blocco di codice:

```
str := ""
x := 9
str += string('0' + x)
str += string('0' + (x-3))
str += string('0' + (x-5))
fmt.Println(str)
```

genera l'output: `964`

2) Sia `str` una varibile di tipo `string`, il seguenti blocco di codice:

```
str := ""
```

```
x := 9
str = string('0' + x) + str
str = string('0' + (x-3)) + str
str = string('0' + (x-5)) + str
fmt.Println(str)
```

genera l'output: `469`

3) Sia `c` una varibile di tipo `rune`, il seguenti blocco di codice:

```
c = '7'
var x int = int(c - '0')
fmt.Println(x)
```

genera l'output: `7`

Esempio di esecuzione

```
> go run seqInversa.go
Inserisci i numeri:
3
5
-7
27
7
152
9
-6
0
Termine inserimento.
Sequenza letta:
3 5 7 9
Sequenza in ordine inverso:
9 7 5 3
Valori da calcolare:
-4
4
```

Esercizio 8 – Testo invertito

Scrivete un programma che legga un testo formato da un numero variabile di righe e lo ristampi dall'ultimo carattere al primo.

SUGGERIMENTI:

Per leggere un numero variabile di righe di testo, dove ogni riga è una stringa in cui compaiono caratteri spazio e terminata da '\n',si può utilizzare il seguente programma:

```
import (
    "bufio"
    "fmt"
    "os"
)

func main() {

    str := ""
    fmt.Println("Introduci una o più righe di testo:")
    scanner := bufio.NewScanner(os.Stdin)
    for scanner.Scan() {

        input := scanner.Text()
        if input == "" {
            fmt.Println("input == \"\": lettura terminata.")
            break
        } else {
            str += input + "\n"
        }

    }
    fmt.Println("Righe di testo lette: ")
    fmt.Print(str)
}
```

Esercizio 9 – Gioco di dadi tra 5 giocatori

Scrivete un programma che chiede all'utente di inserire da linea di comando un numero `t` e quindi simula una partita di dadi tra `5` giocatori e con `t` turni di gioco.

Precisamente, in ogni turno di gioco ogni giocatore lancia due dadi e ottiene come punteggio la somma dei numeri ottenuti dai due lanci (ad esempio se i due lanci danno come risultato `5` e `3`, il punteggio ottenuto è `8`).

Nel corso del gioco, il programma stampa ogni passo di gioco. Ovvero, per ogni lancio di un giocatore, stampa il turno, il nome (descritto sotto) del giocatore, il risultato dei suoi lanci e il punteggio ottenuto dal giocatore.

I giocatori hanno come nome le lettere `A, B, C, D, E.` I turni di gioco sono numerati da `1` a `t (t` OVVIAMENTE compreso!).

Alla fine di ogni turno il programma stampa il vincitore del turno, ovvero colui che ha ottenuto il punteggio maggiore. Se due giocatori ottengono lo stesso punteggio viene ritenuto come punteggio valido quello di colui che ha lanciato dopo.

Quando i turni di gioco sono finiti, il programma stampa il numero di vittorie di ogni giocatore.

SUGGERIMENTI:

1) L'insieme dei giocatori può essere rappresentato da una stringa "ABCDE", i.e., ogni carattere della stringa rappresenta il corrispondente giocatore.

2) Il vincitore di ogni turno può essere memorizzato in una stringa da aggiornare alla fine di ogni turno. Per esempio, la stringa "ABB", ottenuta alla fine del terzo turno, specifica che il primo turno è stato vinto dal giocatore A, ed i successivi dal giocatore B.

3) Per simulare il lancio di un dado che genera punti da `1` a `6` usate le funzioni del package `math/rand.` Il seguente codice stampa `n` numeri casuali nel range `[0, thr)` (dove `n` e `thr` sono i valori di due variabili di tipo `int` e il range `[0, thr)` contiene valori `val` tali che `0 <= val < thr`). Ad esempio, se `thr=7 e n=13,` il seguente blocco di codice stampa `13` volte numeri casuali `x,` tali che: `0 <= x < 7` (`0` COMPRESO `e 7` ESCLUSO).

```
var x, thr int = 13, 7

rand.Seed(int64(time.Now().Nanosecond()))

for i := 0; i< x; i++{
	fmt.Println(rand.Intn(thr))
}
```

Esempi di funzionamento (in grassetto il valore inserito dall'utente):

```
>go run dadi.go
Inserisci il numero di turni di gioco: 7
Turno 1) Giocatore A, lanci di valore: 5 e 5
Turno 1) Giocatore B, lanci di valore: 2 e 5
Turno 1) Giocatore C, lanci di valore: 2 e 3
Turno 1) Giocatore D, lanci di valore: 1 e 5
Turno 1) Giocatore E, lanci di valore: 5 e 3
FINE TURNO 1 - GIOCATORE VINCITORE: A
```

```
Turno 2) Giocatore A, lanci di valore: 1 e 4
Turno 2) Giocatore B, lanci di valore: 5 e 3
Turno 2) Giocatore C, lanci di valore: 4 e 1
Turno 2) Giocatore D, lanci di valore: 5 e 4
Turno 2) Giocatore E, lanci di valore: 4 e 1
FINE TURNO 2 - GIOCATORE VINCITORE: D
Turno 3) Giocatore A, lanci di valore: 2 e 4
Turno 3) Giocatore B, lanci di valore: 2 e 5
Turno 3) Giocatore C, lanci di valore: 1 e 4
Turno 3) Giocatore D, lanci di valore: 1 e 2
Turno 3) Giocatore E, lanci di valore: 2 e 2
FINE TURNO 3 - GIOCATORE VINCITORE: B
Turno 4) Giocatore A, lanci di valore: 5 e 3
Turno 4) Giocatore B, lanci di valore: 3 e 1
Turno 4) Giocatore C, lanci di valore: 4 e 2
Turno 4) Giocatore D, lanci di valore: 3 e 2
Turno 4) Giocatore E, lanci di valore: 5 e 1
FINE TURNO 4 - GIOCATORE VINCITORE: A
Turno 5) Giocatore A, lanci di valore: 5 e 3
Turno 5) Giocatore B, lanci di valore: 5 e 4
Turno 5) Giocatore C, lanci di valore: 1 e 2
Turno 5) Giocatore D, lanci di valore: 3 e 2
Turno 5) Giocatore E, lanci di valore: 2 e 2
FINE TURNO 5 - GIOCATORE VINCITORE: B
Turno 6) Giocatore A, lanci di valore: 3 e 5
Turno 6) Giocatore B, lanci di valore: 2 e 2
Turno 6) Giocatore C, lanci di valore: 5 e 5
Turno 6) Giocatore D, lanci di valore: 5 e 3
Turno 6) Giocatore E, lanci di valore: 4 e 4
FINE TURNO 6 - GIOCATORE VINCITORE: C
Turno 7) Giocatore A, lanci di valore: 4 e 3
Turno 7) Giocatore B, lanci di valore: 4 e 3
Turno 7) Giocatore C, lanci di valore: 3 e 5
Turno 7) Giocatore D, lanci di valore: 3 e 4
Turno 7) Giocatore E, lanci di valore: 2 e 4
```

```
FINE TURNO 7 - GIOCATORE VINCITORE: C
Nome e numero vittorie di ogni giocatore:
A -> 2
B -> 2
C -> 2
D -> 1
E -> 0

>go run dadi.go
Inserisci il numero di turni di gioco: 3
Turno 1) Giocatore A, lanci di valore: 2 e 4
Turno 1) Giocatore B, lanci di valore: 5 e 4
Turno 1) Giocatore C, lanci di valore: 4 e 4
Turno 1) Giocatore D, lanci di valore: 5 e 5
Turno 1) Giocatore E, lanci di valore: 5 e 4
FINE TURNO 1 - GIOCATORE VINCITORE: D
Turno 2) Giocatore A, lanci di valore: 1 e 4
Turno 2) Giocatore B, lanci di valore: 1 e 5
Turno 2) Giocatore C, lanci di valore: 3 e 3
Turno 2) Giocatore D, lanci di valore: 2 e 3
Turno 2) Giocatore E, lanci di valore: 3 e 5
FINE TURNO 2 - GIOCATORE VINCITORE: E
Turno 3) Giocatore A, lanci di valore: 5 e 4
Turno 3) Giocatore B, lanci di valore: 4 e 1
Turno 3) Giocatore C, lanci di valore: 2 e 4
Turno 3) Giocatore D, lanci di valore: 1 e 5
Turno 3) Giocatore E, lanci di valore: 1 e 1
FINE TURNO 3 - GIOCATORE VINCITORE: A
Nome e numero vittorie di ogni giocatore:
A -> 1
B -> 0
C -> 0
D -> 1
E -> 1
```