Laboratorio di algoritmi e strutture dati

Docente: Violetta Lonati

Esercizi di riscaldamento da svolgere a casa (ripasso su cicli, slice, funzioni in Go)

1 Supera 100

Scrivere un programma `supera100.go` che legge da standard input una sequenza di interi positivi terminata da -1 e stampa il primo numero che supera 100, se presente; altrimenti stampa "nessun numero maggiore di 100".

NOTA: Se avete difficoltà a scrivere il programma, potete partire da qui:
`http://parsons.problemsolving.io/puzzle/e334652c5f7f4a82ad21565a01e7cbb7`

2 Conto corrente

Scrivere un programma che legge da riga di comando un intero che rappresenta il saldo di un conto corrente. Il programma legge poi da standard input una serie di numeri interi che rappresentano spese da addebitare sul conto e stampa il saldo finale. La lettura si interrompe quando il saldo è <=0.

NOTA: Se avete difficoltà a scrivere il programma, potete partire da qui:
`http://parsons.problemsolving.io/puzzle/5bc3cba95a44449fb6212e3f5e588504`

3 Andamento

Data da standard input una serie di interi positivi terminata da 0, stampare '+' ogni volta che il nuovo valore è maggiore o uguale al precedente e '-' altrimenti.

NOTA: Se avete difficoltà a scrivere il programma, potete partire da qui:
`http://parsons.problemsolving.io/puzzle/17724c92454744a8bafd7ffd2b889b85`

4 Elaborazione di serie di interi

Scrivete le tre funzioni specificate sotto. Per testarle potete utilizzare il main riportato qui (con gli opportuni adattamenti).

```go
package main

import "fmt"

func main() {
  const N = 10
  numeri := make([]int, N)

  for i := 0; i < N; i++ {
    fmt.Scan(&numeri[i])
  }

  fmt.Println(nomeFunzione(numeri))
}
```

1. Scrivere una funzione `stranoProdotto(numeri []int) int` che, data come parametro una slice di interi, trovi quelli che sono maggiori di 7 e multipli di 4 e ne restituisca il prodotto. Ad esempio, se la slice contiene i numeri 12, 3, 4, 8, 9, 2, la funzione dovrà restituire il numero 96 (pari al prodotto di 12 per 8).

2. Scrivere una funzione `pariDispari(numeri []int)` che, data come parametro una slice di interi, stampi, per ciascun numero, se è pari o dispari.

3. Scrivere una funzione `minDispari(numeri []int) int` che, data una slice di interi, restituisca il più piccolo numero dispari (la slice può contenere sia numeri positivi che negativi).

5 Elaborazione di stringhe

Scrivete le funzioni specificate sotto. Per testarle potete utilizzare il main riportato qui, con gli opportuni adattamenti

```go
package main

import "fmt"

func main() {
  const N = 10
  parole := make([]string, N)

  for i := 0; i < N; i++ {
    fmt.Scan(&parole[i])
  }

  fmt.Println(nomeFunzione(parole))
}
```

1. Scrivere una funzione `quanteConA(parole []string) int` che, data una slice di stringhe, restituisca quante stringhe contengono il carattere 'a'.
2. Scrivere una funzione `primaConA(parole []string) string` che, data una slice di stringhe, restituisca la prima parola che contentiene il carattere 'a?, o la stringa vuota.
3. Scrivere una funzione `piuCorta(parole []string) int` che, data una slice di stringhe, restituisca la lunghezza della stringa più corta in termini di byte.

6 Istogramma a anagrammi

1. Scrivete un programma che legga una riga di caratteri terminata da a-capo, poi legga un'altra lettera e stampi quante volte la lettera compare nella prima riga.

2. Scrivete un programma che legga una riga di caratteri terminata da a-capo e che visualizzi un istogramma con una barra per ogni carattere dell'alfabeto, lunga quanto il numero delle sue occorrenze. Il programma non deve visualizzare le barre delle lettere che non compaiono e non deve fare distinzione fra maiuscole e minuscole (a tal fine potete usare le funzioni dichiarate nel file `ctype.h`).

   Esempio di funzionamento:

```
C'era un RAGAZZO che come ME amava i Beatles e I rolling StoneS.
A *******
B *
C ***
E ********
G **
H *
I ***
L ***
M ***
N ***
O ****
R ***
S ***
T **
U *
V *
Z **
```

3. Due parole costituiscono un *anagramma* se l'una si ottiene dall'altra permutando le lettere (es: attore, teatro). Scrivete un programma che legga due parole e verifichi se sono anagrammi.

   Suggerimento: potete sfruttare il programma scritto per l'esercizio precedente.

7 Prime vocali

Si vuole scrivere un programma che legga una sequenza di stringhe e stampi, per ogni stringa, la sua prima vocale (per semplicità assumiamo che le stringhe contengano solo lettere minuscole). Nel caso in cui una stringa non contenga vocali il programma stampa "la parola non contiene vocali".

8 Inizia con 'a', inizia con 'b'

Si vuole scrivere un programma che legga una sequenza di dieci stringhe e stampi il numero di stringhe che cominciano con la lettera `a` e il numero di stringhe che cominciano con `b`.
Per calcolare il numero di stringhe che cominciano per `a`, usiamo il piano del *conteggio*; lo stesso piano serve anche a calcolare il numero di stringhe che cominciano per `b`.

9 Elettricità

Vogliamo leggere una sequenza di N interi (almeno 3), che descrivono il consumo di elettricità in N giorni, e stampare i giorni in cui il consumo è stato inferiore rispetto sia al giorno prima che al giorno dopo. I giorni sono numerati a partire da 1.

10 Coppie di numeri uguali

Scrivere un programma che legga da standard input una sequenza di numeri terminata da zero e conti quante sono le coppie di numeri naturali consecutivi uguali.

11 Massimo numero vocali

Si vuole scrivere una funzione che, presa come argomento una slice di stringhe, restituisca il numero massimo di consonanti contenute in una stringa (assumiamo per semplicità che le stringhe contengano solo caratteri minuscoli).

12 Massimo numero di zeri

Scrivere un programma `massimo_n_zeri.go` che legge da standard input una sequenza di interi terminata da -1 e stampa il numero che contiene il maggior numero di 0. Nel caso in cui vi siano più numeri che contengono il massimo numero di 0, il programma stampa l?ultimo incontrato. Ad esempio, se la sequenza letta è `3040 145 80 1707 105002 78 1970 6005 -1` il programma stampa `105002`.

13 Somme delle sottosequenze crescenti

Scrivere un programma che legge una sequenza di interi e stampa la somma degli interi in ciascuna sottosequenza crescente. Il programma interrompe la lettura quando riceve un segnale di EOF. Ad esempio, su input `1 2 13 0 7 8 9 -1 0 2` il programma stampa le somme 16, 24 e 1.