FILTRO: filtro.go

Creare un programma **robot.go** nel quale viene definito il tipo **Robot** contenente le coordinate **x** e **y** del robot (interi) e la **direzione** (un tipo a piacere) che assumerà un valore corrispondente alla direzione del robot (su/giù/destra/sinistra)

Definire (inferendone il prototipo) la funzione **costruisci** che, dati due interi ed una stringa, restituisca una variabile di tipo **Robot** ed un valore booleano. Se tutti i dati passati sono corretti, il valore booleano ritornato sarà **true**, e la variabile di tipo **Robot** restituita avrà la cordinata **x** corrispondente al primo valore passato, la coordinata **y** corrispondente al secondo valore passato, e la **direzione** stabilita in base alla stringa passata come terzo valore. Specificatamente, se la stringa è "su", "giù", "destra" o "sinistra", la direzione del robot avrà il valore corrispondente. Se la stringa è diversa dai valori specificati sopra, il valore booleano restituito sarà falso. In questo caso non ci interessa come sarà la variabile robot. Es:

```
r2d2,ok := costruisci(5,5,"su") //ok sarà true
c3po,ko := costruisci(0,0,"rotto") //ko sarà false
```

Definire (inferendone il prototipo) la funzione **stato** che, dato un parametro di tipo **Robot** stampi le sue coordinate e la sua direzione. Si assuma che solo i Robot costruiti correttamente verranno passati. Es:

```
stato(r2d2)
```

restituirà:

```
{5 5 su}
```

Definire (inferendone il prototipo) la funzione **avanza** che, dato un puntatore a un **Robot** (si assuma corretto), faccia avanzare il robot nella direzione corrente di una posizione. Si assuma che la coordinata x vada da sinistra verso destra e la coordinata y da su verso giù. Quindi se la direzione del robot è su, avanzare decrementerà la coordinata y di 1. Es:

```
avanza(&r2d2)
stato(r2d2)
```

restituirà:

```
{5 4 su}
```

Definire (inferendone il prototipo) la funzione **ruota** che, dato un puntatore a un **Robot** (si assuma corretto) ed un valore booleano che specifichi il senso della rotazione (se true orario, altrimenti antiorario), giri il robot nel senso indicato. Es:

```
ruota(&r2d2, True)
stato(r2d2)
```

restituirà:

```
{5 4 destra}
```

Griglia

Estendere il programma **robot.go** definendo la funzione **griglia** che, data una slice di **Robot** (si assuma corretti e non sovrapposti), e quattro valori interi (xmin, xmax, ymin, ymax), mostri a schermo una griglia che si estende tra le coordinate x e y minime e massime specificate (incluse). Per ciascuna casella nella quale non vi è un robot presente verrà stampato il carattere "□" (valore intero 9633). Per ciascuna casella nella quale vi è presente un robot verrà stampato un carattere che dipenderà dalla direzione del robot: ⬆ (valore intero 11014) se su, ⬇ (valore intero 11015) se giù, ⬅ (valore intero 11013) se sinistra, ➡ (valore intero 10145) se destra. Es:

```
r1,_ := costruisci(1,1,"destra")
r2,_ := costruisci(2,2,"giù")
r3,_ := costruisci(2,5,"su")
r4,_ := costruisci(2,7,"su")

robots := []Robot{r1,r2,r3,r4}

griglia(robots, 0, 4, 0, 3)
```

restituirà:

```
□□□□□
□➡□□□
□□⬇□□
□□□□□
```

Vista

Estendere il programma **robot.go** definendo la funzione **vista** che, data una slice di robot (si assuma corretti e non sovrapposti), restituisca

una slice contenente tutte le coppie di robot che si vedono tra di loro (le coppie sono slice di due elementi). Due robot si vedono tra di loro se sono sulla stessa linea (hanno la stessa coordinata x o y) e sono direzionati uno nel verso dell'altro. ATTENZIONE: un robot vede tutti i robot nella stessa linea verso i quali è direzionato e che sono direzionati verso di lui. Ad esempio, considerando la slice robots dell'esercizio precedente:

```
c := vista(robots)
fmt.Println(c)
```

stamperà qualcosa di questo genere (Nota: non è richiesta la stampa perfetta ma solo che la slice restituita sia corretta):

```
[[{2 2 giù} {2 5 su}] [{2 2 giù} {2 7 su}]]
```

Nuovi Robot

Estendere il programma **robot.go**, scrivendo la funzione main in modo tale che richieda all'utente di inserire dei nuovi robot da standard input, uno alla volta. Il robot dovrà essere inserito specificando le coordinate x e y come interi, e la direzione come stringa. I valori possibili per la direzione sono su/giù/destra/sinistra. Qualora un robot sia già presente alle coordinate specificate, il nuovo robot non dovrà essere inserito, il sistema scriverà "occupato" e chiederà un nuovo inserimento. Inserendo un valore di direzione non valido si terminerà l'input. Infine, dovrà essere stampata la lista dei robot inseriti, es (le righe indentate sono inserite dall'utente):

```
Inserisci un robot:
    2 2 su
Inserisci un robot:
    0 1 destra
Inserisci un robot:
    0 1 destra
occupato
Inserisci un robot:
    0 0 fine
Robot inseriti:
{0 1 destra}
{2 2 su}
```