

```

var s string
func Scan (&s)
/* PROTOCOLS: // HOST / ... */

```

```

(1)
var i, pos1, pos2 int
for i=0; i < len(s); i++ {
    if s[i] == '/' {
        break
    }
}
pos1 = i
(2)
for i=pos1+2; i < len(s); i++ {
    if s[i] == '/' {
        break
    }
}
pos2 = i
host := s[pos1+2 : pos2]
prot := s[: pos1]

```

import "strings"

① pos1 := strings.IndexRune(s, '/')

② pos2 := strings.IndexRune(s[pos1+2:], R) + pos1 + 2

---

s = "http://pippo/qualcosa"  
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 ...

pos1 = 5

s[pos1+2:] = "pippo/qualcosa"  
0 1 2 3 4 5 6 7 8 9 ...

unicode

IsLower

```
var s string
func Scan (&s)
c := 0
for i := 0; i < len(s); i++ {
    if s[i] >= 'a' &&
       s[i] <= 'z' {
        c++
    }
}
}
```

C++

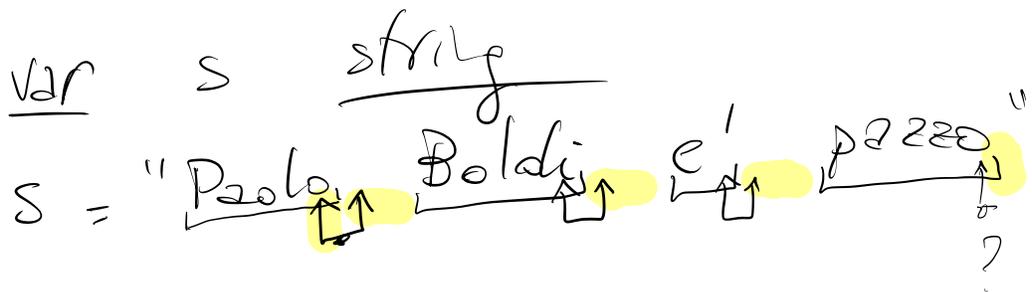
```
for -, r := range s {  
  if r >= 'a' &&  
    r <= 'z' {  
    c++  
  }  
}
```

```
for -, r := range s {  
  if unicode.IsLower(r) {  
  }  
  c++  
}
```

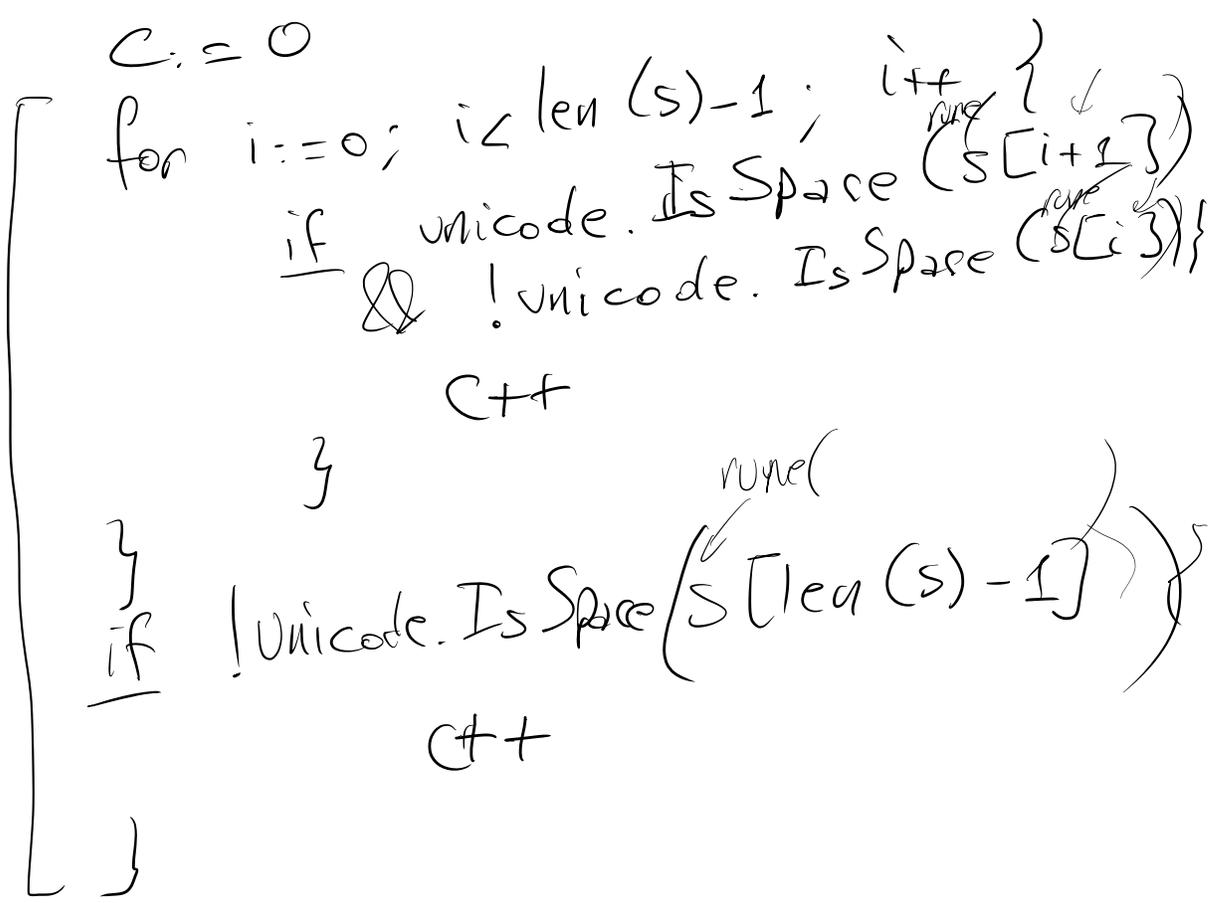
- DATA UNA STRINGA ASCII

CONTA DA QUANTE  
PAROLE E' COMPOSTA

```
var s string
s = "Paolo Boldi e' pazzo"
```



```
c := 0
for i := 0; i < len(s)-1; i++ {
    if unicode.IsSpace(s[i+1])
        && !unicode.IsSpace(s[i])
        c++
}
if !unicode.IsSpace(s[len(s)-1])
    c++
}
```



```

prev := rune('L')
for _, r := range s {
    if unicode.IsSpace(r) &&
       !unicode.IsSpace(prev) {
        C++
    }
    prev = r
}
if !unicode.IsSpace(prev) {
    C++
}

```

# SELEZIONE MULTIPLA

switch {  
    espressione (selettore) } // true

case espressione, espressione, ...  
|||  
|||

case espressione, espressione, ...  
|||  
|||

case espressione, espressione, ...  
|||  
|||

default :  
}

```
var n int
...
switch n * n }
case 1, 16, 27 :
    a = 3
case 9 :
    a = 4
default :
    a = 0
}
```

switch {

case  $x > 4$ ,  $x == 2$ :

$z = 3$ ;

case  $y > 5$ :

$z = 7$

default :

$z = 9$

}

---

if  $x > 4$  ||  $x == 2$  {  
 $z = 3$

} else if  $y > 5$  {  
 $z = 7$

} else {  
 $z = 9$

}

- DATO UN NUMERO TRA  
1 E 99 DETERMINARE  
LA STRINGA CORRISP.  
(32 → "threntwo")  
[FUNZIONE]

```

func numero In Lettere (n int) string {
  if n >= 10 && n <= 19 {
    switch n {
      case 10:
        return "dieci"
      case 11:
        return "undici"
      :
      case 19:
        return "dieciannove"
    }
  }
}

```

```

}
var res, fin string
switch n/10 {
  case 2:
    res = "venti"
    fin = "i"
  case 3:
    res = "trenta"
    fin = "a"
}

```

```

    ...
    case 9:
        res = "nove"
        fin = "2"
}
n%10 {
    case 1:
        res += "one"
    case 2:
        res += "two"
    ...
    case 9:
        res += "nove"
}
}
return res
}

```

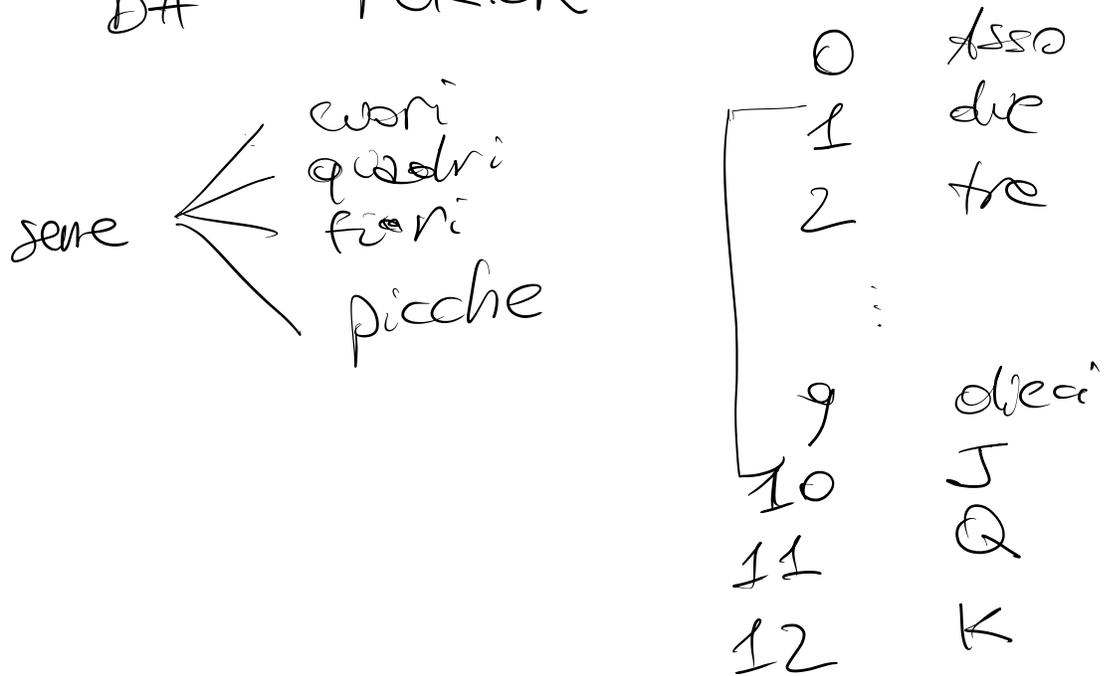
A circled asterisk (\*) is connected by an arrow to the word switch.

⊗

```
[ if  
  }  
]
```

```
n % 10 != 1 && n % 10 != 8  
res += fin
```

- STAMPARE UNA CARTA  
DA POKER



0 → 12	carte di cuori
13 → 25	" " quadri
26 → 38	" " fiori
39 → 51	" " picche

---

A ♡      2 ♠  
J ♣

func stapa CardsPoker (cards int)

switch cards/13 {  
    case 0:           "♥"  
        sue =  
    case 1:           "♦"  
        sue =

    ...  
} switch cards/13 {  
    case 0:           val = "A"  
    case 10:          val = "J"  
    case 11:          val = "Q"  
    case 12:          val = "K"

default:

val = fmt.Sprintf("%d",  
                  cards/13)

}  
return val + " " + some  
}

(cards (1 10) T1)