count $\begin{cases} = 0 & \text{copiando} \\ > 0 & \text{contando} \end{cases}$

c ——— carattere appena letto

count \ c	*	non *
= 0	count = 1	output (c)
> 0	count++	output (count) output (c) count = 0

```
func convert(s string) (t string) {
    count := 0
    for i:=0; i<len(s); i++ {
        if count==0 && s[i]=='*' {
            count = 1
        } else if count==0 {
            t += string(rune(s[i]))
        } else if s[i] == '*' {
            count++
        } else {
            t += strconv.Itoa(count)
            t += string(rune(s[i]))
            count = 0
        }
    }
    if count>0 {
        t += strconv.Itoa(count)
    }
}
```

ESERCIZI

- Scrivete una funzione che dato un numero in base 2 (sottoforma di stringa) restituisca il valore corrispondente. Testatela contro quella di libreria su 1000 stringhe binarie generate a caso.
- Dato un numero segreto (scritto intercalando caratteri non-numerici a caratteri numerici) calcolarne il quadrato.

  ##3ab17c#ccc#0#

  ↓

  3170^2
- Data una stringa s e una stringa t, stampare in quali posizioni di t compare s

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

s = banana banana bananaz

t = nan

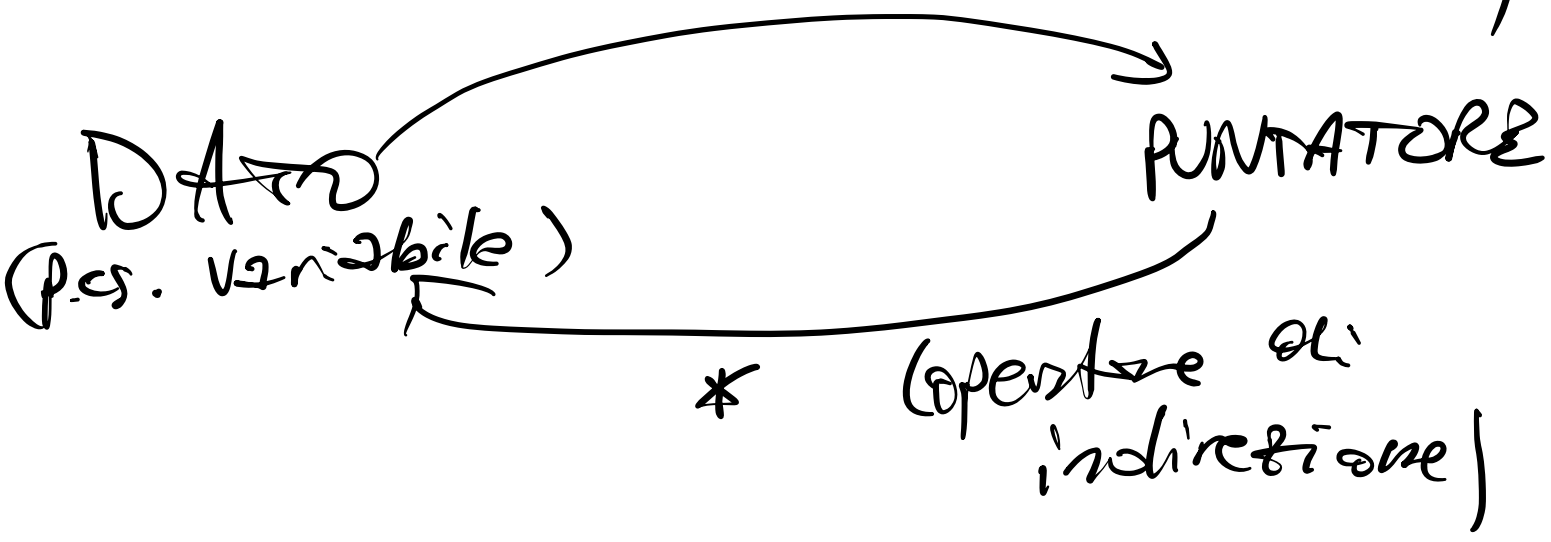
2
8
16

PUNTATORI

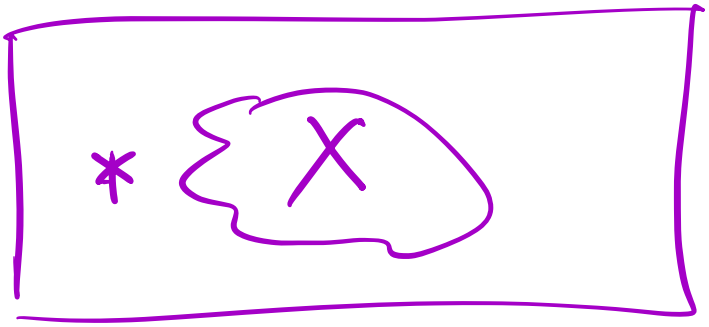
```
func f(y int) {
    y++
}

func main() {
    x := 7
    f(x)
    f(x*5)
    fmt.Println(x)
}
```

```
func main() {
    var n int
    fmt.Scan(&n)
}
```

PUNTATORE = LOCAZIONE DI MEMORIA DOVE SI TROVA UN DATO



PUNTATORI HANNO UN TIPO DI BASE



X è un tipo

tipo "puntatore a X"

var x int

var p *int

var q *string

var r **int

var t ****string

VALORE SPECIALE: nil (ZERO)

var s string

var p *string

s = "pippo"

p = &s

fmt.Println(s)

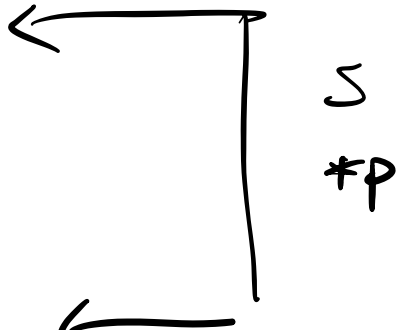
fmt.Println(p)

fmt.Println(*p)

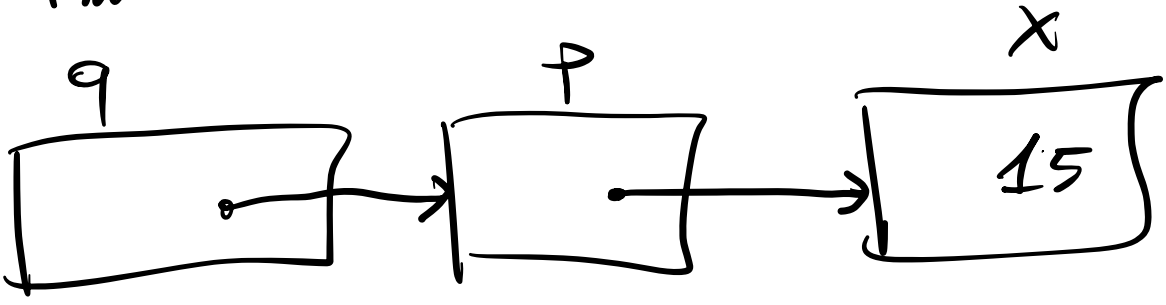
fmt.Println(len(*p))

fmt.Println((*p)[1])

*p = "ciao"



```
var   x   int
var   p   *int
var   q   **int
x = 15
p = &x
q = &p
(**q)++
fmt.Println(x)
```



var x int

var p *int

var q **int

x = 7

p = &x

q = &p

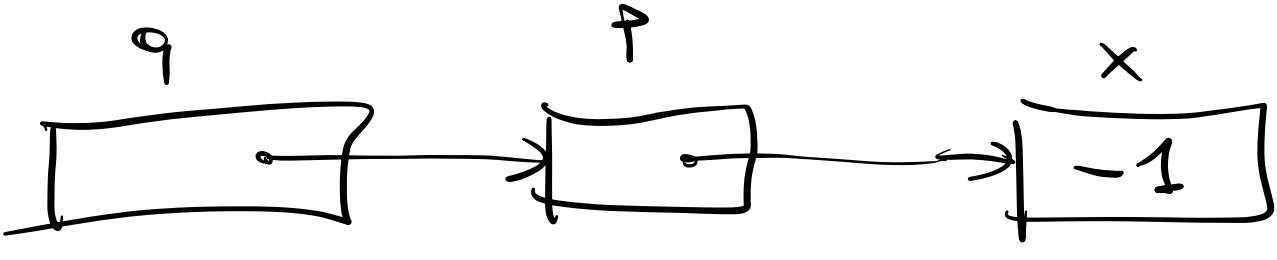
*p = 50

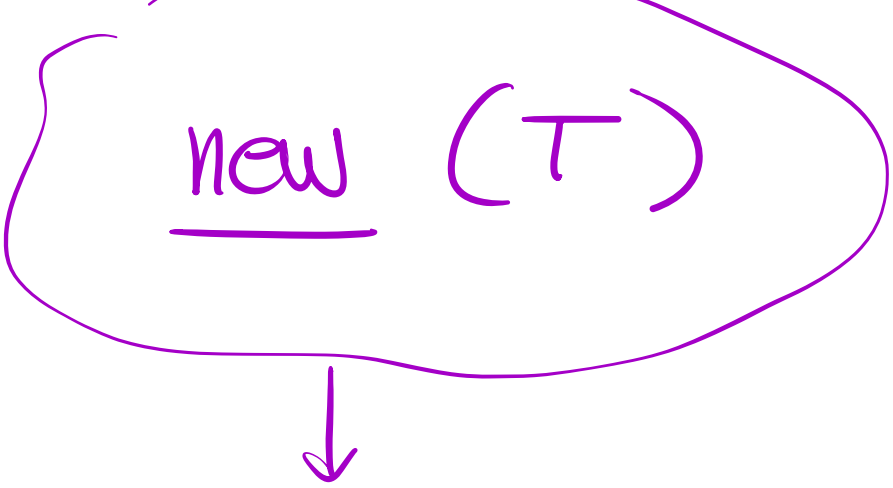
**q = *p - (x+1)

(*p)++

(**q)--

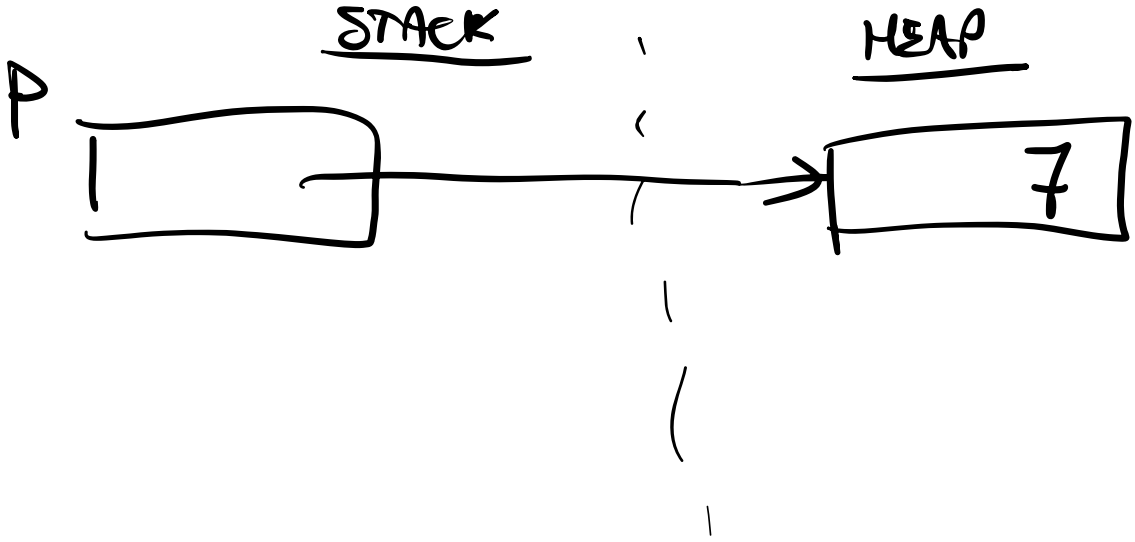
fmt.Println(x)





ALLOCA NELLO HEAP
UNO SPAZIO DI MEMORIA
PER IL TIPO T E
NE RESTITUISCE
LA LOCAZIONE

Var p *int
p = new (int)
*p = 7
fmt. Println (*p)



var x int

var p *int

var q **int

→ x = 7

→ p = &x

→ q = &p

→ p = new (int)

→ *p = 50

→ q = new (*int)

→ *q = &x

→ **q = 5

→ *q = new (int)

→ **q = 12

fmt.Println (x, *p, **q)

5 50 12

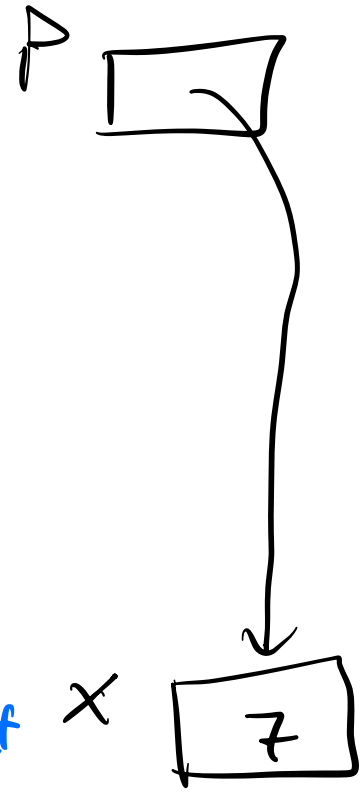
→ p = &x

→ *p ++

→ fut. Printf(n (x, *p, **q)

6 6 12

```
func f(p *int) {
    (*p)++
}

func main() {
    x := 7
    f(&x)
    fmt.Println(x)
}
```



var q *int
q = &x
f(q)

```
func f(y int) int {
    return y+1
}
func main() {
    x := 7
    x = f(x)
    fmt.Println(x)
}
```

5