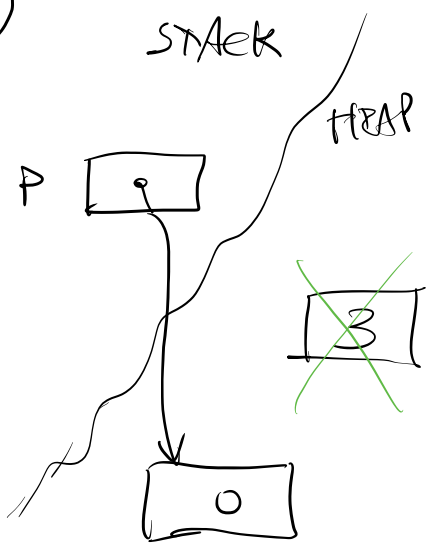
var p *int
var x int
p = &x
*p = 3

*p ALIAS x

---

new (T)

var p *int
p = new (int)
*p = 3
p = new (int)



STRUCT

PREMESSA

type {ID} {tipo}

TYPE DEFINITION

SOGGETTIVE

type pf **float64

var x, y, z pf

pippo.go

```
package main
import (
...
)
type pf **float64
var i int
func pippo (x int) {
    var a, b, c pf
    ...
}
func pluto (a, b pf) int {
```

...

}

---

type money float64

var x, y, z money

var f, g float64

x += f * y

x += money(f) * y

type ID = tipo

TYPE ALIAS

type money = float64

var x, y, z money

var f, g float64

x += f * y

struct

VAR s struct {

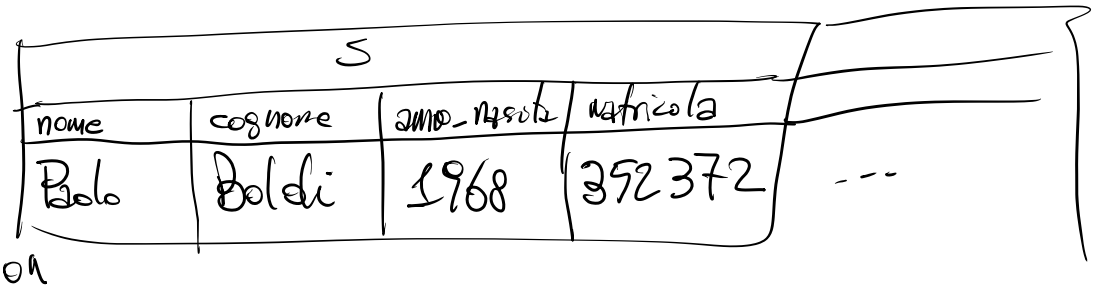
nome, cognome — string

anno_nascita — int

matricola — string

...

}



dot-notation

s.nome = "Bolo"

s.cognome = "Boldi"

s.anno_nascita = 1968

...

```
type studente struct {
    nome, cognome string
    anno_nascita int
    ...
}

var s, t, p studente
var x *studente

x = new(studente)
```

```
type data struct {
    g, m, a int
}
var d1, d2 data
```

(A)
```
d1.g = 29
d1.m = 11
d1.a = 1968
```

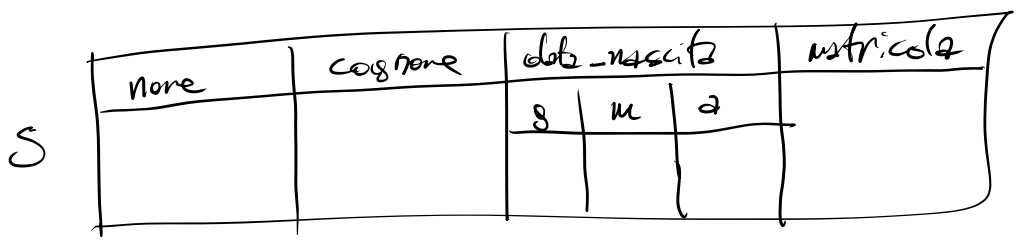
(B)
```
d1.g, d1.m, d1.a = 29, 11, 1968
```

(C)
```
d1 = data {29, 11, 1968}
```
literale struct

(D)
```
d1 = data {g: 29, [illegible] a: 1968}
```

type studente struct {
nome, cognome string
data_nascita data
matricola string
}

var s studente



fmt.Scan(&s.nome)
fmt.Scan(&s.cognome)
fmt.Scan(&s.data_nascita.g)
fmt.Scan(&s.data_nascita.m)
fmt.Scan(&s.data_nascita.a)
...

s = studente {nome: "Paolo", cognome: "Boldi"
data_nascita: data {g: 29, m: 11, a: 1968},
matricola: "352 372"}

- FUNZIONE CHE DATO UN ANNO RESTITUISCE LA DATA DEL NATALE DI QUELL'ANNO

```
func christmas (y int) data {
    var r data
    r = data{g: 25, m: 12, a: y}
    return r
}
```

- FUNZIONE CHE DATA UNA DATA RESTITUISCE LA DATA CHE CADE 6 MESI DOPO

```
func addHalfYear(d data) data {
    var r data
    r.g = d.g
    if d.m <= 6 {
        r.m = d.m + 6
        r.a = d.a
    } else {
        r.m = d.m + 6 - 12
        r.a = d.a + 1
    }
}
```

(1)

(*)
```
r.g = d.g
r.m = d.m + 6
if r.m > 12 {
        r.m -= 12
        r.g++
}
```

(**)
```
r.m = (d.m + 6 - 1) % 12 + 1
r.g = d.g + (d.m + 6 - 1) / 12
```

- SCRIVERE UNA FUNZIONE CHE DATA UNA DATA LA MODIFICA IN MODO DA PORTARLA ALL'INIZIO DEL MESE SUCCESSIVO

```
func nextMonth (d *data) {
        (*d).g = 1
        (*d).m ++
        if (*d).m > 12 {
                (*d).m = 1
                (*d).g++
        }
}
```

d.g

var d1 = date {11, 11, 2022}
nextMonth (&d1)