

STABILIRE SE UN
NUMERO È PRIMO

var m, d int

fun. Scan (&n)

for d = 2; d < n; d++
if n % d == 0 {

break

}

}
if d < n {
fun. Println ("Composto")

lele J

break

funct. Printk("Primo")

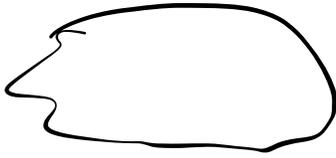
}

ISTRUZIONI break/continue

① break

② continue

① break : interrompe l'esecuzione del ciclo in cui è contenuta

for  {

if 

```
}  
    break  
}
```

```
for ... i < n ... {  
    for ... j < m ... {
```

```
        for ... k < h ... {  
            break  
        }  
        if k < h {  
            break;  
        }
```

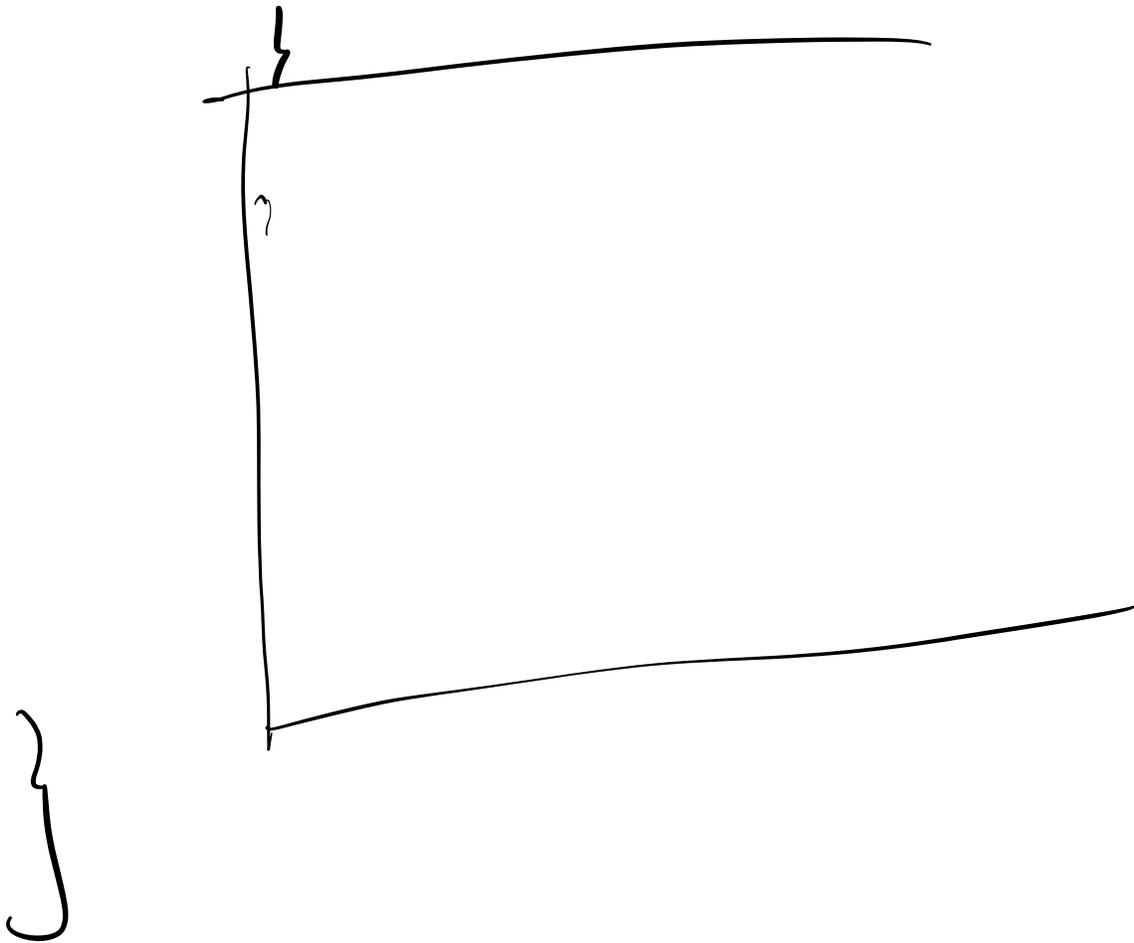
ewersuone
dapidada

```
    }  
    if j < m {  
        break  
    }
```

```
}
```

② continue: forza la prossima esecuzione del ciclo

```
for i = 1; i < n; i++  
  if i % 10 == 0  
    continue }
```



- DATO n STAMPA I NUMERI
PRIMI $\leq n$

var n int

fun. Scan (&n)

for x:=2; x<=n; x++ {

if math.IsPrime(x) {
fun.Println(x)

}

}

var m, d int

fact.Scan (&n)

for x:=2; x<=n; x++}

for d:=2; d<x; d++}
if x%d==0}
break

}
if d==x }

fact.Println(x)

}

}

- STAMPA PRIMI M
NUMERI PRIMI

var m, d, c int

fact. Scan (&n)

for x:=2; ^{c<n} ; x++ }

for d:=2; d<x; d++ }

if x%d == 0 ?
break

}

if

d==x {

fact. Println(x)

C++

~~if c==n }~~

~~break~~

~~}~~

}

}

FUNKION 1

package main

import ...
func isPrime (n int) bool {
 

? func main() {
 var n int

 fmt.Scan (&n)

for x:=2; x<=n; x++ {

if

 isPrime(x) {

 fmt.Println(x)

 }

 }

]

FUNZIONE

HEADER

func

nome
funzione

parametri
formali

tipi
rest.

CORPO DELLA
FUNZIONE

3

FUNZIONE CHE CALCOLA
IL QUADRATO DI UN
NUMERO INTERO

parametri formali

```
func sqr (a int) int {  
  var ris int  
  ris = a * a  
  return ris  
}
```

```
}  
func main() {  
  var n int  
  fmt.Scan(&n)  
  x := sqr(n)  
  fmt.Println(x)  
  x = sqr(5)  
}
```

*parametri
attuali o
argomenti*

$x = \text{sqrt}(v)$
fmt.Println(x)
 $y := \text{sqrt}((x+1)*7)$

fmt.Print(y)

~~$z := \text{sqrt}(x, y)$~~

~~$z := \text{sqrt}()$~~

~~$z := \text{sqrt}(5.0)$~~

func ipotenusa(cat1, cat2 float64)
float64 {

var x float64

$x = \text{cat1} * \text{cat1} + \text{cat2} * \text{cat2}$

return math.Sqrt(x)

```
}  
func main () {  
    x := ipotenus(7.0, 8.5)  
    fmt.Println(x)  
    return math.Sqrt(cat1*cat1  
        + cat2*cat2)  
}
```

- SCRIVETE UNA FUNZIONE
CHE DETERMINA SE
UN INTERO È PARI

```
func isEven (x int) bool }
```

return $x/2 == 0$

}

func main() {

var n int

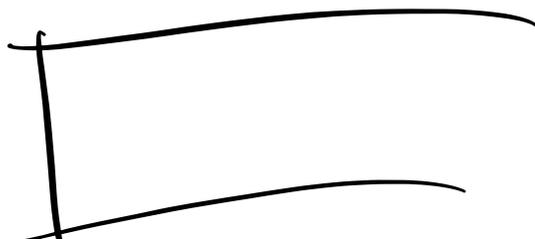
fmt.Scan(&n)

if isEven(n) {
fmt.Println("E pari")

else {
fmt.Println("E dispari")

}

if isEven(n*n + 7) && n > 0 {



ESERCIZI

- Scrivere una funzione che determina se un numero è divisore di un altro
- Scrivere una funzione che calcola quanti zeri capiscono nella scrittura di un numero.
Usatela per calcolare quanti 0 dovete scrivere se scrivete tutti i numeri da 1 a 2

1000000.

- Scrivete una funzione che dato le coordinate

di due punti sul
piano cartesiano restituisce
il coeff. angolare della
retta passante.

BACK TO THE FUTURE

```
func   isPrime (x int) bool {  
    var   d int  
    for   d=2; d<x; d++ }  
        if   x%d==0 {  
            break  
        }  
    }  
    if   d==x { return true  
    }  
    else { return false  
    }  
}
```



```
func isPrime (x int) bool {  
  var d int  
  for d=2; d<x; d++  
    if x%d==0 {  
      return false  
    }  
  }  
  return true  
}
```

```
}
```

```
func isid () {  
  var M int  
  for. Scan (&n)
```

```

for x:=2; x<n; x++
  if isPrime(x)
    fat. Println(x)
  }
}

```

ESERCIZIO

- Un numero è primo di Mersenne se è primo ed è della forma $2^p - 1$ con p primo.

Scrivi una funz. che determini se un numero è primo di

Merse me.

Usala pr sturpe
: prdui n prdui st.

Merse me.