



Laboratorio di  
programmazione - Ed. 1  
- 2023/24

A. Morpurgo

Laboratorio 3

Tecniche per  
l'individuazione di  
errori

Argomenti

I cicli: controllo del  
numero di iterazioni

Problemi tipici

# Laboratorio di programmazione

## Corsi di laurea triennale in Informatica, Informatica musicale e Informatica per la comunicazione digitale

### Turno A (A-Can) - Turno B (Cao-D)

Docenti: Anna Morpurgo - Andrea Trentini  
Tutor: Leonardo Albani - Federico Bruzzzone

Dipartimento di Informatica  
Università degli Studi di Milano

A.A. 2023-2024



Laboratorio di  
program-  
mazione - Ed. 1  
- 2023/24

A. Morpurgo

Laboratorio 3

Tecniche per  
l'individuazione di  
errori

Argomenti

I cicli: controllo del  
numero di iterazioni

Problemi tipici

# Laboratorio 03

26/10/2023



# Strumenti per programmare: la tracciatura

Laboratorio di  
programmazione - Ed. 1  
- 2023/24

A. Morpurgo

Laboratorio 3

Tecniche per  
l'individuazione di  
errori

Argomenti

I cicli: controllo del  
numero di iterazioni

Problemi tipici

## Tracciatura dell'esecuzione di un programma programma

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var n int
7     s := 0
8     for {
9         fmt.Scan(&n)
10        if n == 0 {
11            break
12        }
13        s += n
14    }
15    fmt.Println(s)
16 }
17
18 //tracciare su input: 1 9 7 3 0 5 5
```

input: 1 9 7 3 0 5 5  
TRACING O TRACCIATURA

| n. istr. | s                             | n |
|----------|-------------------------------|---|
| 6        |                               | 0 |
| 7        | 0                             |   |
| 9        |                               | 1 |
| 13       | 1                             |   |
| 9        |                               | 9 |
| 13       | 10                            |   |
| 9        |                               | 7 |
| 13       | 17                            |   |
| 9        |                               | 3 |
| 13       | 20                            |   |
| 9        |                               | 0 |
| 10       | Condizione verificata → break |   |
| 15       | Stampa s, quindi 20           |   |



# Analisi e comprensione vs scrittura

Laboratorio di  
program-  
mazione - Ed. 1  
- 2023/24

A. Morpurgo

Laboratorio 3

Tecniche per  
l'individuazione di  
errori

Argomenti

I cicli: controllo del  
numero di iterazioni

Problemi tipici

## analisi/comprendione

- **livello del testo (lessicale e sintattico):** il codice per come è scritto
- **livello dell'esecuzione del programma (semantico e della macchina concettuale):** il programma mentre è in esecuzione, cosa fa.
- **livello dello scopo:** lo scopo del programma, che compito svolge o che problema risolve

## scrittura di programmi

- **conoscenza lessicale e sintattica:** conoscenza dei termini e della sintassi di Go
- **conoscenza semantica e concettuale:** conoscenza di come funzionano i vari costrutti Go e che effetti producono
- **conoscenza strategica:** capacità di applicare la conoscenza sintattica e concettuale di Go per risolvere problemi



# Il costrutto for: controllo del numero di iterazioni

Laboratorio di  
programmazione - Ed. 1  
- 2023/24

A. Morpurgo

Laboratorio 3

Tecniche per  
l'individuazione di  
errori

Argomenti

I cicli: controllo del  
numero di iterazioni

Problemi tipici

- con contatore (*counter-controlled iteration*): **for ternario**  
for  $i := 0; i < n; i++\{$   
    blocco di istruzioni  
}
- con condizione (*condition controlled iteration*): **for unario**  
for <condizione>{  
    blocco di istruzioni  
}
- senza condizione (*infinite loop*): **for zerario**  
for { // o for ; ; { o for true {  
    blocco di istruzioni  
    if <condizione>{  
        break  
    }  
    blocco di istruzioni  
}



# Il costrutto for: controllo del numero di iterazioni

Laboratorio di  
programmazione - Ed. 1  
- 2023/24

A. Morpurgo

Laboratorio 3

Tecniche per  
l'individuazione di  
errori

Argomenti

I cicli: controllo del  
numero di iterazioni

Problemi tipici

Con contatore (*counter-controlled iteration*): **for ternario**

- **Livello *sintattico*:**

```
for i:= 0; i < n; i++){  
    blocco di istruzioni  
}
```

- **Livello *semantico (notional machine)*:**

```
for Ai := 0; Bi < 10; Ci++ {  
    Dfmt.Println(i)  
}
```

A solo la prima volta, poi B, D, C,  
B, D, C, ..., B, fino a che B risulta falsa

1. inizializza i a 0
2. verifica se  $i < n$
3. se sì, esegui blocco istruzioni, incrementa i, torna a 2.
4. se no, vai alla prima istruzione dopo la chiusa graffa

- **Livello *strategico*:**

- esegui una stessa azione  $n$  volte
- elabora una serie di  $n$  valori
- elabora i valori da  $j$  a  $k$  (o dal  $j$ -esimo a  $k$ -esimo di una serie)



# Il costrutto for: controllo del numero di iterazioni

Laboratorio di  
program-  
mazione - Ed. 1  
- 2023/24

A. Morpurgo

Laboratorio 3  
Tecniche per  
l'individuazione di  
errori  
Argomenti  
I cicli: controllo del  
numero di iterazioni  
Problemi tipici

Con condizione (*condition controlled iteration*): **for unario**

- **Livello *sintattico*:**

```
for <condizione>{  
    blocco di istruzioni  
}
```

- **Livello *semantico* (*notional machine*):**

1. verifica la condizione
2. se vera, esegui blocco istruzioni, torna a 1.
3. se falsa, vai alla prima istruzione dopo la chiusa graffa

- **Livello *strategico*:** elabora fino a ottenere un certo risultato



# Il costrutto for: controllo del numero di iterazioni

Laboratorio di  
program-  
mazione - Ed. 1  
- 2023/24

A. Morpurgo

Laboratorio 3

Tecniche per  
l'individuazione di  
errori

Argomenti

I cicli: controllo del  
numero di iterazioni

Problemi tipici

Senza interruzione (*infinite loop*): **for zerario**

- **Livello *sintattico*:**

```
for { // o for ; ; { o for true {  
    [blocco di istruzioni 1]  
    if <condizione>{  
        break  
    }  
    [blocco di istruzioni 2]  
}
```

- **Livello *semantico* (*notional machine*):**

1. esegui blocco istruzioni 1
2. verifica la condizione
3. se vera, vai alla prima istruzione dopo la chiusa graffa
4. se falsa, esegui blocco istruzioni 2 e torna a 1

- **Livello *strategico*:** elabora fino a incontrare un certo  
valore o situazione





# break e continue

Laboratorio di  
program-  
mazione - Ed. 1  
- 2023/24

A. Morpurgo

Laboratorio 3

Tecniche per  
l'individuazione di  
errori

Argomenti

I cicli: controllo del  
numero di iterazioni

Problemi tipici

```
1 for i:= 0; i < n; i++){  
2     block of instructions  
3     if <condizione>{  
4         break           //exit the for loop, go to line 8  
5     }  
6     another block  
7 }  
8
```



# break e continue

Laboratorio di  
program-  
mazione - Ed. 1  
- 2023/24

A. Morpurgo

Laboratorio 3

Tecniche per  
l'individuazione di  
errori

Argomenti

I cicli: controllo del  
numero di iterazioni

Problemi tipici

```
1 for i:= 0; i < n; i++){
2     block of instructions
3     if <condizione>{
4         continue    // go to line 1, i.e. to the next loop,
                       // skipping "another block"
5     }
6     another block
7 }
8
```



# Progettazione di un ciclo

Laboratorio di  
program-  
mazione - Ed. 1  
- 2023/24

A. Morpurgo

Laboratorio 3

Tecniche per  
l'individuazione di  
errori

Argomenti

I cicli: controllo del  
numero di iterazioni

Problemi tipici

- blocco di istruzioni da ripetere
- controllo dell'iterazione: condizione o indice (e passo)
- uso di break e continue



# Problemi e schemi per l'iterazione

Laboratorio di  
programmazione - Ed. 1  
- 2023/24

A. Morpurgo

Laboratorio 3

Tecniche per  
l'individuazione di  
errori

Argomenti

I cicli: controllo del  
numero di iterazioni

Problemi tipici

Ci sono **problemi** o compiti di base che richiedono l'uso dell'iterazione e che sono **molto comuni**, sia in sé che come sottoproblemi di problemi più complessi.

Data una serie di valori:

- calcolo di un **totale** (sommatoria, produttoria, ecc) sui valori della serie
- **conteggio** di elementi della serie
- ricerca del **min/max** della serie
- **ricerca lineare** di un valore nella serie
- **verifica di una proprietà** degli elementi di una serie
- elaborazione di **valori adiacenti** (fibonacci, ecc.) nella serie
- **ripetizione** di una stessa azione o sequenza di azioni sugli elementi della serie

Vedremo che la soluzione di ciascuno di questi problemi ha una sua struttura tipica.