

Strutture di controllo

Programmazione strutturata

Ramificazione

Introduzione all'iterazione

Liceo G.B. Brocchi - Bassano del Grappa (VI)
Liceo Scientifico - opzione scienze applicate
Giovanni Mazzocchin

Ancora sulla ramificazione

- Scriviamo un programma che si comporta in questo modo:
 - se l'utente del programma inserisce il carattere 'a' da stdin, il programma stampa 'A' su stdout
 - se l'utente del programma inserisce il carattere 'b' da stdin, il programma stampa 'B' su stdout
 - se l'utente del programma inserisce il carattere 'c' da stdin, il programma stampa 'C' su stdout
 - se l'utente del programma inserisce il carattere 'd' da stdin, il programma stampa 'D' su stdout
 - con qualsiasi altro input, il programma stamperà su stdout il messaggio `input not accepted, please enter a, b, c, or d`

Un programma logicamente sbagliato

```
int main() {
    char choice;
    printf("enter a character: ");
    scanf("%c", &choice);
    if (choice == 'a') {
        putchar('A');
        putchar('\n');
    }
    if (choice == 'b') {
        putchar('B');
        putchar('\n');
    }
    if (choice == 'c') {
        putchar('C');
        putchar('\n');
    }
    if (choice == 'd') {
        putchar('D');
        putchar('\n');
    }
    else {
        puts("input not accepted, please enter a, b, c, or d");
    }
}
```

**cosa succede se l'utente
inserisce 'a', oppure 'b',
oppure 'c', oppure 'd'?**

Un programma logicamente sbagliato

```
cyofanni@LAPTOP-I0S1RKRC:~/Desktop/high-school-cs-class/c_lectures/starter$ ./if_else_if
enter a character: a
A
input not accepted, please enter a, b, c, or d
cyofanni@LAPTOP-I0S1RKRC:~/Desktop/high-school-cs-class/c_lectures/starter$ ./if_else_if
enter a character: b
B
input not accepted, please enter a, b, c, or d
cyofanni@LAPTOP-I0S1RKRC:~/Desktop/high-school-cs-class/c_lectures/starter$ ./if_else_if
enter a character: c
C
input not accepted, please enter a, b, c, or d
cyofanni@LAPTOP-I0S1RKRC:~/Desktop/high-school-cs-class/c_lectures/starter$ ./if_else_if
enter a character: d
D
cyofanni@LAPTOP-I0S1RKRC:~/Desktop/high-school-cs-class/c_lectures/starter$
```

Cos'è successo?

```
if (choice == 'a') {  
    putchar('A');  
    putchar('\n');  
}  
if (choice == 'b') {  
    putchar('B');  
    putchar('\n');  
}  
if (choice == 'c') {  
    putchar('C');  
    putchar('\n');  
}  
if (choice == 'd') {  
    putchar('D');  
    putchar('\n');  
}  
else {  
    puts("input not accepted, please enter a, b, c, or d");  
}
```

**l'utente inserisce 'a'. Il flusso di controllo del programma arriva alla valutazione della condizione puntata dalla freccia.
NB: dobbiamo ancora entrare nel blocco dell'if**

Cos'è successo?

```
if (choice == 'a') {  
    putchar('A');  
    putchar('\n');  
}  
if (choice == 'b') {  
    putchar('B');  
    putchar('\n');  
}  
if (choice == 'c') {  
    putchar('C');  
    putchar('\n');  
}  
if (choice == 'd') {  
    putchar('D');  
    putchar('\n');  
}  
else {  
    puts("input not accepted, please enter a, b, c, or d");  
}
```

la condizione del primo if viene valutata true.

Il flusso di controllo del programma passa alle istruzioni del blocco del primo if, che vengono eseguite tutte

Cos'è successo?

```
if (choice == 'a') {  
    putchar('A');  
    putchar('\n');  
}  
if (choice == 'b') {  
    putchar('B');  
    putchar('\n');  
}  
if (choice == 'c') {  
    putchar('C');  
    putchar('\n');  
}  
if (choice == 'd') {  
    putchar('D');  
    putchar('\n');  
}  
else {  
    puts("input not accepted, please enter a, b, c, or d");  
}
```

il flusso di controllo del programma arriva alla valutazione della condizione puntata dalla freccia, che è falsa.
NB: dobbiamo ancora entrare nel blocco dell'if

Cos'è successo?

```
if (choice == 'a') {  
    putchar('A');  
    putchar('\n');  
}  
if (choice == 'b') {  
    putchar('B');  
    putchar('\n');  
}  
if (choice == 'c') {  
    putchar('C');  
    putchar('\n');  
}  
if (choice == 'd') {  
    putchar('D');  
    putchar('\n');  
}  
else {  
    puts("input not accepted, please enter a, b, c, or d");  
}
```

la condizione viene valutata false. Il blocco del secondo if non viene eseguito

Cos'è successo?

```
if (choice == 'a') {  
    putchar('A');  
    putchar('\n');  
}  
if (choice == 'b') {  
    putchar('B');  
    putchar('\n');  
}  
if (choice == 'c') {  
    putchar('C');  
    putchar('\n');  
}  
if (choice == 'd') {  
    putchar('D');  
    putchar('\n');  
}  
else {  
    puts("input not accepted, please enter a, b, c, or d");  
}
```

il flusso di controllo del programma arriva alla valutazione della condizione puntata dalla freccia.

NB: dobbiamo ancora entrare nel blocco dell'if

Cos'è successo?

```
if (choice == 'a') {  
    putchar('A');  
    putchar('\n');  
}  
if (choice == 'b') {  
    putchar('B');  
    putchar('\n');  
}  
if (choice == 'c') {  
    putchar('C');  
    putchar('\n');  
}  
if (choice == 'd') {  
    putchar('D');  
    putchar('\n');  
}  
else {  
    puts("input not accepted, please enter a, b, c, or d");  
}
```

la condizione viene valutata false. Il blocco del terzo if non viene eseguito

Cos'è successo?

```
if (choice == 'a') {  
    putchar('A');  
    putchar('\n');  
}  
if (choice == 'b') {  
    putchar('B');  
    putchar('\n');  
}  
if (choice == 'c') {  
    putchar('C');  
    putchar('\n');  
}  
if (choice == 'd') {  
    putchar('D');  
    putchar('\n');  
}  
else {  
    puts("input not accepted, please enter a, b, c, or d");  
}
```

il flusso di controllo arriva alla valutazione della condizione del quarto if. La condizione viene valutata false. Il blocco del quarto if non viene eseguito

Cos'è successo?

```
if (choice == 'a') {  
    putchar('A');  
    putchar('\n');  
}  
if (choice == 'b') {  
    putchar('B');  
    putchar('\n');  
}  
if (choice == 'c') {  
    putchar('C');  
    putchar('\n');  
}  
if (choice == 'd') {  
    putchar('D');  
    putchar('\n');  
}  
else {  
    puts("input not accepted, please enter a, b, c, or d");  
}
```

il blocco dell'else viene eseguito solo se la condizione dell'if precedente era falsa. Effettivamente era falsa, perché 'a' != 'd', quindi il blocco dell'else viene eseguito!

Una versione logicamente corretta

```
int main() {
    char choice;
    printf("enter a character: ");
    scanf("%c", &choice);
    if (choice == 'a') {
        putchar('A');
        putchar('\n');
    }
    else {
        if (choice == 'b') {
            putchar('B');
            putchar('\n');
        }
        else {
            if (choice == 'c') {
                putchar('C');
                putchar('\n');
            }
            else {
                if (choice == 'd') {
                    putchar('D');
                    putchar('\n');
                }
                else {
                    puts("input not accepted: please enter a, b, c or d");
                }
            }
        }
    }
}
```

il blocco `else` puntato dalla freccia blu viene eseguito solo se la condizione dell'`if` puntato dalla freccia verde è falsa

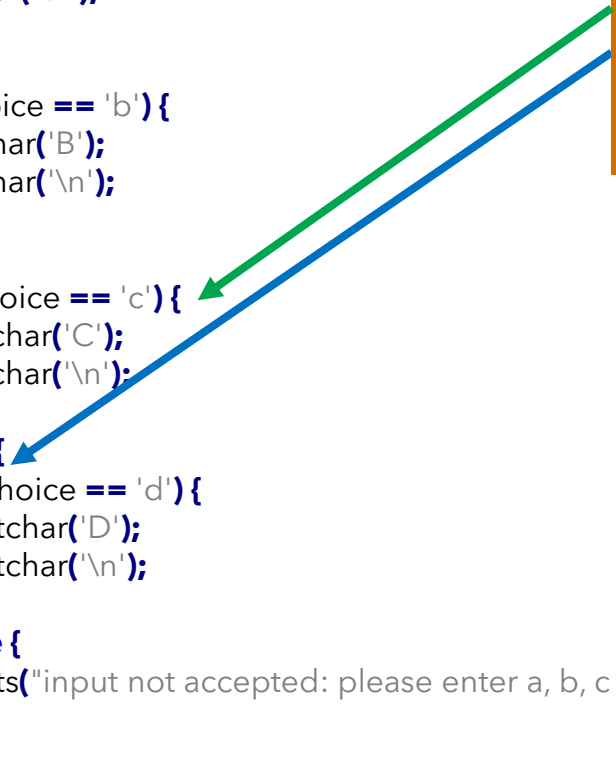
Una versione logicamente corretta

```
int main() {
    char choice;
    printf("enter a character: ");
    scanf("%c", &choice);
    if (choice == 'a') {
        putchar('A');
        putchar('\n');
    }
    else {
        if (choice == 'b') {
            putchar('B');
            putchar('\n');
        }
        else {
            if (choice == 'c') {
                putchar('C');
                putchar('\n');
            }
            else {
                if (choice == 'd') {
                    putchar('D');
                    putchar('\n');
                }
                else {
                    puts("input not accepted: please enter a, b, c or d");
                }
            }
        }
    }
}
```

il blocco `else` puntato dalla freccia blu viene eseguito solo se la condizione dell'`if` puntato dalla freccia verde è falsa

Una versione logicamente corretta

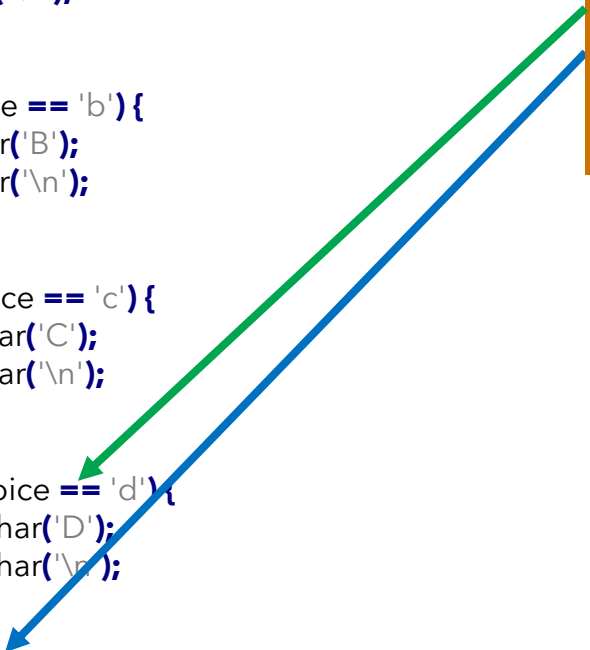
```
int main() {  
    char choice;  
    printf("enter a character: ");  
    scanf("%c", &choice);  
    if (choice == 'a') {  
        putchar('A');  
        putchar('\n');  
    }  
    else {  
        if (choice == 'b') {  
            putchar('B');  
            putchar('\n');  
        }  
        else {  
            if (choice == 'c') {  
                putchar('C');  
                putchar('\n');  
            }  
            else {  
                if (choice == 'd') {  
                    putchar('D');  
                    putchar('\n');  
                }  
                else {  
                    puts("input not accepted: please enter a, b, c or d");  
                }  
            }  
        }  
    }  
}
```



il blocco `else` puntato dalla freccia blu viene eseguito solo se la condizione dell'`if` puntato dalla freccia verde è falsa

Una versione logicamente corretta

```
int main() {  
    char choice;  
    printf("enter a character: ");  
    scanf("%c", &choice);  
    if (choice == 'a') {  
        putchar('A');  
        putchar('\n');  
    }  
    else {  
        if (choice == 'b') {  
            putchar('B');  
            putchar('\n');  
        }  
        else {  
            if (choice == 'c') {  
                putchar('C');  
                putchar('\n');  
            }  
            else {  
                if (choice == 'd') {  
                    putchar('D');  
                    putchar('\n');  
                }  
                else {  
                    puts("input not accepted: please enter a, b, c or d");  
                }  
            }  
        }  
    }  
}
```



il blocco `else` puntato dalla freccia blu viene eseguito solo se la condizione dell'`if` puntato dalla freccia verde è falsa

Sintassi alternativa per evitare l'annidamento

```
int main() {
    char choice;
    printf("enter a character: ");
    scanf("%c", &choice);
    if (choice == 'a') {
        putchar('A');
        putchar('\n');
    }
    else if (choice == 'b') {
        putchar('B');
        putchar('\n');
    }
    else if (choice == 'c') {
        putchar('C');
        putchar('\n');
    }
    else if (choice == 'd') {
        putchar('D');
        putchar('\n');
    }
    else {
        puts("input not accepted: please enter a, b, c or d");
    }
}
```

Una nuova struttura di controllo

- Immaginate un programma che deve stampare una riga di 10 asterischi per 15 volte, con la funzione `printf`
- Con le nostre conoscenze attuali, dovremmo mettere in **sequenza** 15 istruzioni che chiamano la funzione `printf`
- È noiosissimo scrivere programmi in questo modo. Immaginate di dover mettere in sequenza non 15, ma 15 000 istruzioni così...

Una nuova struttura di controllo

```
#include <stdio.h>
```

```
int main() {  
    printf("*****");  
    printf("*****");  
    printf("*****");  
    printf("*****");  
    printf("*****");  
    printf("*****");  
    printf("*****");  
    printf("*****");  
    printf("*****");  
    printf("*****");  
    printf("*****");  
    printf("*****");  
    printf("*****");  
    printf("*****");  
    printf("*****");  
}
```

**Programmare così è una sofferenza...
Sicuramente per realizzare
programmi poco più che banali ci
serve qualcos'altro**

**Sarebbe utile avere una struttura di
controllo che ci permette di dire al
computer: fai una cosa un certo
numero di volte**

Una nuova struttura di controllo

- Immaginate un programma che deve stampare la tabellina di un numero letto da standard input... per ora potremmo scriverlo così:

```
int main() {  
    int number;  
    printf("enter an integer number: ");  
    scanf("%d", &number);  
  
    puts("the times table of %d is:\n", number);  
    printf("%d\t", number * 1);  
    printf("%d\t", number * 2);  
    printf("%d\t", number * 3);  
    printf("%d\t", number * 4);  
    printf("%d\t", number * 5);  
    printf("%d\t", number * 6);  
    printf("%d\t", number * 7);  
    printf("%d\t", number * 8);  
    printf("%d\t", number * 9);  
    printf("%d\n", number * 10);  
}
```

Una nuova struttura di controllo

- Una versione alternativa potrebbe essere la seguente...

```
int number;
printf("enter an integer number: ");
scanf("%d", &number);
int mult = 1;
printf("the times table of %d is:\n", number);
printf("%d\t", number * mult);
mult++;
printf("%d\t", number * mult);
mult++;
printf("%d\t", number * mult);
mult++;
printf("%d\t", number * mult);
mult++;
printf("%d\t", number * mult);
mult++;
printf("%d\t", number * mult);
mult++;
printf("%d\t", number * mult);
mult++;
printf("%d\t", number * mult);
mult++;
printf("%d\n", number * mult);
```

l'istruzione `mult++` equivale all'istruzione: `mult = mult + 1`
`++` è chiamato operatore di incremento

Iterazione

- Abbiamo bisogno di chiedere al computer di eseguire un blocco di istruzioni un certo numero di volte
- Ci serve una struttura di controllo chiamata **iterazione**
- Nei linguaggi C-like, questa struttura di controllo può essere realizzata tramite la keyword `while`
- La sintassi è analoga a quella dell'costrutto `if`

```
if (boolean condition) {  
}  
while (boolean condition) {  
}
```

Iterazione

```
while (boolean condition) {  
  
}
```

Possiamo tradurlo così in italiano:

**fintantoché la condizione «boolean condition» è vera,
esegui le istruzioni contenute nel blocco delimitato
dalle graffe**

Iterazione

- **while** (2 == 2) {}
- **while** (4 < 5);
- **while** (1) {}
- **while** (0) {}

cosa succede quando il flusso di controllo del programma arriva a queste istruzioni?

Le tabelline

```
/*  
the program prints the times table of  
an integer number read from standard input.  
*/
```

```
int main() {  
    int number, multiplier;  
    multiplier = 1;  
    printf("%s", "enter an integer, the program will show its times table: ");  
    scanf("%d", &number);  
    while (multiplier <= 10) {  
        printf("%d\t", number * multiplier);  
        multiplier++;  
    }  
    printf("\n");  
}
```

**quanto vale multiplier
una volta usciti dal ciclo?**

Le tabelline... codice meno leggibile

```
/*  
the program prints the times table of  
an integer number read from standard input.  
*/
```

```
int main() {  
    int number, multiplier;  
    multiplier = 0;  
    printf("%s", "enter an integer, the program will show its times table: ");  
    scanf("%d", &number);  
    while (multiplier < 10) {  
        printf("%d\t", number * (multiplier + 1));  
        multiplier++;  
    }  
    printf("\n");  
}
```

**quanto vale multiplier
una volta usciti dal ciclo?**

Drills

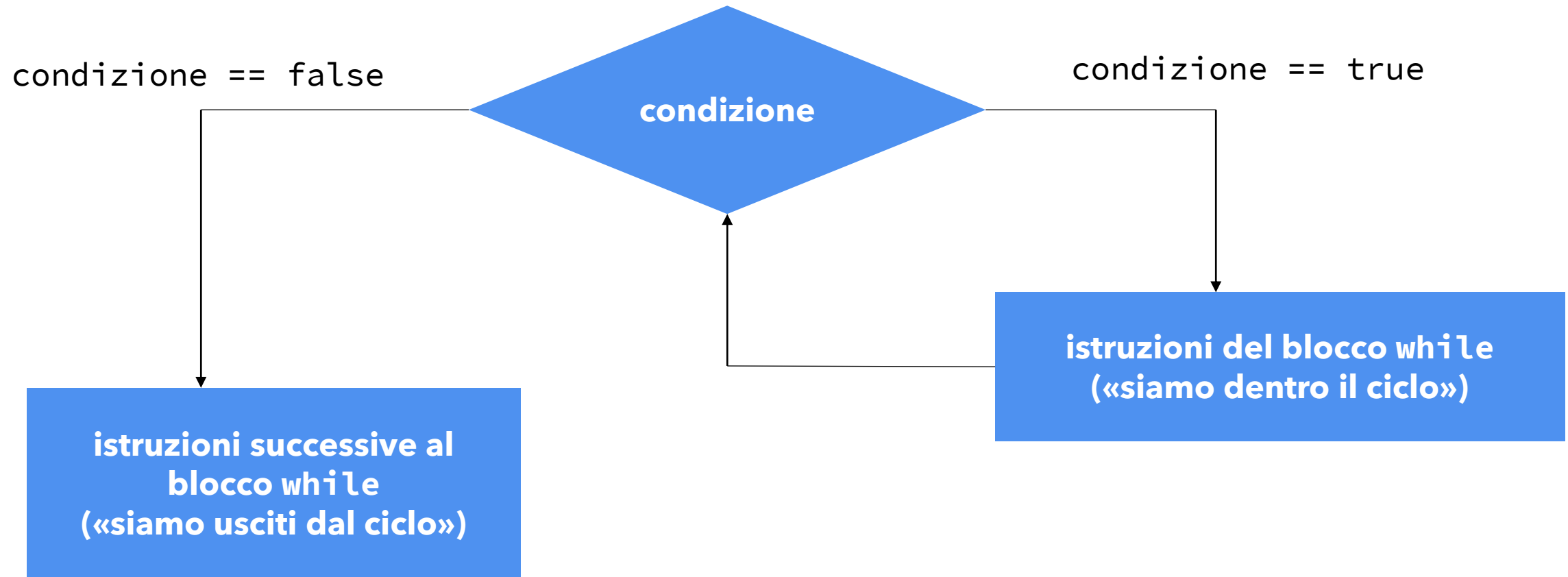
```
int k = 0;  
  
while (k < 10) {  
    k += 1;  
}
```

```
int k = 10;  
  
while (k > 0) {  
    k -= 1;  
}
```

Drills

```
int k = 1;  
int s = 0;  
  
while (k <= 100) {  
    s += k;  
    k += 1;  
}
```

Iterazione – diagramma di flusso



Iterazione – valore sentinella (*flag*)

/*

the program computes the sum of a sequence of numbers read from stdin.

When the user enters the flag -1, the program prints the sum.

*/

Riflettete prima di lanciarsi nella scrittura del codice.

Quante variabili serviranno?

Quali strutture di controllo serviranno?

Iterazione – valore sentinella (*flag*)

/*

the program computes the product of a sequence of numbers read from stdin.

When the user enters the flag -1, the program prints the product.

*/

Riflettete prima di lanciarsi nella scrittura del codice.

Quante variabili serviranno?

Quali strutture di controllo serviranno?

Teorema di Boehm-Jacopini

Qualsiasi algoritmo può essere realizzato tramite le seguenti 3 strutture di controllo:

- **sequenza**
- **selezione** (detta anche **ramificazione**)
- **iterazione**