

# Corso sul linguaggio C++

## Modulo 2

### 2 – Funzioni ricorsive

## Prerequisiti

- Programmazione elementare in C++
- Utilizzo di funzioni con parametri

# Introduzione

In questa Unità vediamo un particolare utilizzo delle funzioni in C++, che consiste nello scrivere funzioni **ricorsive**.

Per scrivere una funzione ricorsiva occorre utilizzare uno schema che viene descritto nell'Unità attraverso alcuni esempi.

# Argomenti

- Lo schema ricorsivo
- Il calcolo di  $a * b$
- L'istanza di funzioni ricorsive
- Il calcolo di  $n!$
- I numeri di Fibonacci

# Informazioni generali

Di un problema spesso se ne può dare una **definizione iterativa** o una **definizione ricorsiva**.

Quando si scrive la funzione che risolve il problema:

- la *definizione iterativa* prevede che la funzione utilizzi le s.d.c. iterative (cicli)
- la *definizione ricorsiva* non prevede cicli all'interno della funzione ma risolve il problema chiamando se stessa, riducendo il problema, via via, a casi sempre più semplici, fino a giungere alla formulazione più elementare di questo.

# Lo schema ricorsivo

Per scrivere una funzione ricorsiva, è necessario individuare:

- 1) il **nome della funzione**
- 2) una **condizione** che stabilisca se ci troviamo nel caso più elementare (*condizione di fine ricorsione*)
- 3) il caso più elementare della funzione (**passo base**)
- 4) il caso generale in cui la funzione istanzia se stessa (**passo induttivo**).

Se il passo induttivo è una espressione, allora:

- se l'istanza si trova all'inizio di essa si ha **ricorsività in testa**
- se l'istanza si trova alla fine di essa si ha **ricorsività in coda**
- se l'istanza avviene più di una volta si ha **ricorsività multipla**

## Il calcolo di $a * b$

La funzione **prodotto** (A,B) di due interi A e B può essere implementata come detto poiché si osserva che:

$$\begin{aligned} \text{prodotto}(A,B) &= A * B = A && \text{(se } B=1) \\ \text{prodotto}(A,B) &= A * B = A + A * (B-1) = A + \text{prodotto}(A,B-1) && \text{(se } B>1) \\ & \underbrace{\hspace{1.5cm}}_{\text{prodotto}} \quad \underbrace{\hspace{1.5cm}}_{\text{prodotto}} \end{aligned}$$

La definizione prevede due casi: il caso elementare (B=1) per cui  $A * B = A$  e il caso  $B > 1$  in cui il **prodotto**(A,B) viene definito tramite **prodotto**(A,B-1) ossia mediante se stesso.

## Il calcolo di $a * b$

```
#include <iostream>
#include <cstdlib>
using namespace std;
int prodotto (int, int);
int main(int argc, char *argv[])
{ int a, b;
  cout<<"Immetti il primo fattore: ";   cin>>a;
  cout<<"Immetti il secondo fattore: "; cin>>b;
  cout<<"Il prodotto "<<a<<" * "<<b<<" vale "<<prodotto(a, b)<<endl;
  system("Pause");
  return 0;
}
int prodotto (int a, int b)
{ if (b==1) return a;
  else return a+prodotto(a, b-1);
}
```

Condizione

Passo base

Passo induttivo  
(ricorsione in coda)

ATTIVITA': scrivere un'applicazione ricorsiva per il calcolo del prodotto di due interi  $a$  e  $b$ .

## Il calcolo di $a * b$

Eseguendo la traccia per la funzione si avrebbe, ad esempio:

$$\begin{aligned} \text{prodotto}(3,4) &= 3 + \text{prodotto}(3,3) = \\ &= 3 + \text{prodotto}(3,2) = \\ &= 3 + \text{prodotto}(3,1) \\ &= 3 \end{aligned}$$

## L'istanza di funzioni ricorsive

Vogliamo ora accennare a come sia possibile che, in fase di esecuzione della funzione, possano essere presenti più **istanze in contemporanea** della stessa funzione.

Quando avviene una istanza di una funzione ricorsiva, all'interno di un'altra istanza (o del programma chiamante), interviene la pila di attivazione nella quale vengono memorizzati:

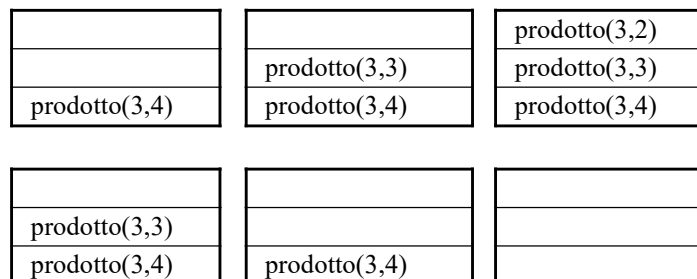
- il **punto di ritorno**
- i valori dei **parametri attuali**
- i valori delle **variabili locali**

## L'istanza di funzioni ricorsive

In questo modo, quando una istanza termina, l'istanza che l'ha chiamata, riassume esattamente lo stato che aveva al momento della chiamata (valori dei parametri attuali e delle variabili locali) e riprende la sua esecuzione esattamente dal punto in cui era stata interrotta.

## L'istanza di funzioni ricorsive

Come si vede dalle sequenze, la pila di attivazione accoglie le istanze che via via vengono sospese. La riattivazione di ciascuna istanza avviene in ordine inverso a quello di sospensione. Al termine la pila risulta vuota.



# Il calcolo di $n!$

```
#include <iostream>
#include <cstdlib>
using namespace std;
int fatt_ricors (int n);
int main()
{ int n;
  cout<<"Immetti il numero: ";
  cin>>n;
  cout<<"Il fattoriale di "<<n<<" e:"<<fatt_ricors(n)<<endl;
  system("Pause");
  return EXIT_SUCCESS;
}
int fatt_ricors (int n)
{ if (n==0) return (1);
  else return (n*fatt_ricors(n-1));
}
```

**ATTIVITA'**: scrivere un'applicazione ricorsiva per il calcolo del fattoriale di un intero  $n$ .

Condizione

Passo base

Passo induttivo  
(ricorsione in coda)

M.Malatesta 2- Funzioni ricorsive-06

13  
17/10/2008

# I numeri di Fibonacci

```
#include <iostream>
#include <cstdlib>
using namespace std;
int fibonacci (int i);
int main(int argc, char *argv[])
{ int n;
  cout<<"Quale numero della serie si vuole conoscere?";
  cin>>n;
  cout<<"Il numero vale "<<fibonacci(n)<<endl;
  system("Pause");
  return 0;
}
int fibonacci (int i)
{ if (i<=1) return i;
  else return fibonacci(i-1)+fibonacci(i-2);
}
```

**ATTIVITA'**: scrivere un'applicazione ricorsiva per il calcolo del valore dell' $n$ -esimo numero di Fibonacci

Condizione

Passo base

Passo induttivo  
(ricorsione multipla)

M.Malatesta 2- Funzioni ricorsive-06

14  
17/10/2008

## Cosa ho imparato

- Quali sono le caratteristiche di una funzione ricorsiva e qual è il suo funzionamento
- Quali sono gli elementi necessari per poter scrivere una funzione ricorsiva
- I vari tipi di ricorsività

## Cosa ho imparato a fare

- Individuare gli elementi di una funzione ricorsiva
- Scrivere semplici applicazioni che facciano uso di funzioni ricorsive
- Individuare un procedimento ricorsivo per risolvere un problema
- Trasformare un procedimento ricorsivo in iterativo e viceversa



## Terminologia

- Ricorsività
- Ricorsione in testa
- Ricorsione in coda
- Ricorsione multipla
- Passo base
- Passo induttivo

## Altre fonti di informazione

- A.Lorenzi-A.Rizzi - Il linguaggio C++ - ed. ATLAS
- G.Callegarin – Informatica e programmazione (v. unico) - CEDAM