POLITECNICO DI MILANO

Programmazione ricorsiva

Fondamenti di Informatica

Daniele Loiacono

Ricorsione

- Che cos’è la ricorsione?
  - Un sottoprogramma P richiama se stesso (ricorsione diretta)
  - Un sottoprogramma P richiama un’altro sottoprogramma Q che comporta un’altra chiamata a P (ricorsione indiretta)
- A cosa serve?
  - È una tecnica di programmazione molto potente
  - Permette di risolvere in maniera elegante problemi complessi

Programmazione ricorsiva

- Per risolvere un problema attraverso la programmazione ricorsiva sono necessari alcuni elementi
  - **Caso base**: caso elementare del problema che può essere risolto immediatamente
  - **Passo ricorsivo**: chiamata ricorsiva per risolvere uno o più problemi più semplici
  - **Costruzione della soluzione**: costruzione della soluzione sulla base del risultato delle chiamate ricorsive

Esempio: il fattoriale

- Definizione:
  f(n) = n! = n*(n-1)*(n-2)*…*3*2*1
- Passo ricorsivo:
  f(n) = n*f(n-1)
- Caso base:
  f(0)=1

```
int factRic(int n)
{
  if (n==0)
      return 1;
  else
      return n*factRic(n-1);
}
```

Funzionamento ricorsione

```
int ris = factRic(3);
```

```
?
ris
```

Funzionamento ricorsione

```
int ris = factRic(3);
```

?
ris

n:3

```
int factRic(int n){
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

Funzionamento ricorsione

```
int ris = factRic(3);
```

? ris

```
int factRic(int n){        n:3
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

```
int factRic(int n){        n:2
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

Funzionamento ricorsione

```
int ris = factRic(3);
```

ris: ?

n:3
```
int factRic(int n){
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

n:2
```
int factRic(int n){
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

n:1
```
int factRic(int n){
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

Funzionamento ricorsione

```
int ris = factRic(3);
```

?
ris

```
int factRic(int n){        n:3
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

```
int factRic(int n){        n:2
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

```
int factRic(int n){        n:1
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

```
int factRic(int n){        n:0
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

Funzionamento ricorsione

```
int ris = factRic(3);
```

?

ris

n:3

```
int factRic(int n){
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

n:2

```
int factRic(int n){
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

n:1

```
int factRic(int n){
  if (n==0)
    return 1;
  else
    return n*1;}
```

Funzionamento ricorsione

```
int ris = factRic(3);
```

?

ris

```
int factRic(int n){
  if (n==0)
    return 1;
  else
    return n*factRic(n-1);}
```

n:3

```
int factRic(int n){
  if (n==0)
    return 1;
  else
    return n*1;}
```

n:2

Funzionamento ricorsione

```
int ris = factRic(3);
```

?

ris

n:3

```
int factRic(int n){
  if (n==0)
    return 1;
  else
    return n*2;}
```

Funzionamento ricorsione

```
int ris = factRic(3);
```

$\boxed{6}$

ris

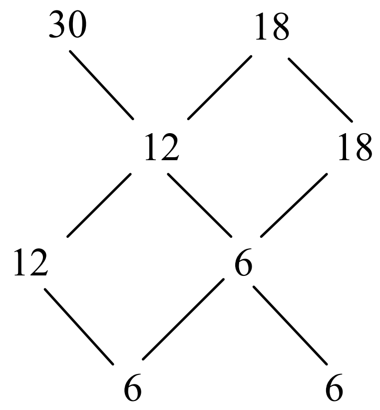
Esempio: Fibonacci

- È una sequenza di numeri interi in cui ogni numero si ottiene sommando i due precedenti nella sequenza. I primi due numeri della sequenza sono per definizione pari ad 1.
  - $f_1 = 1$ (caso base)
  - $f_2 = 1$ (caso base)
  - $f_n = f_{n-1} + f_{n-2}$ (passo ricorsivo)

```
int fiboRic(int n)
{
  if (n==1 || n==2)
        return 1;
  else
        return fiboRic(n-2)+fiboRic(n-1);
}
```

Esempio: MCD

- Algoritmo di Euclide
  - se m = n, MCD(m,n) = m (caso base)
  - se m > n, MCD(m,n) = MCD(m-n,n) (caso risorsivo)
  - se m < n, MCD(m,n) = MCD(m,n-m) (caso risorsivo)
- Esempio: MCD(30,18)



Esempio: MCD

- Algoritmo di Euclide
  - se m = n, MCD(m,n) = m (caso base)
  - se m > n, MCD(m,n) = MCD(m-n,n) (caso risorsivo)
  - se m < n, MCD(m,n) = MCD(m,n-m) (caso risorsivo)
- Implementazione

```
int MCDeuclidRic(int m, int n)
{
  if (m==n)
      return m;
  else if (m>n)
      return MCDeuclidRic(m-n,n);
  else
      return MCDeuclidRic(m,n-m);
}
```

Problemi nell’uso della ricorsione

- Terminazione della catena ricorsiva
  - È presente il caso base?
  - Viene raggiunto sempre dalla catena di chiamate ricorsive?
  - Esempi

```
int factRic(int n)
{
  return n*factRic(n-1);
}
```

Catena infinita di chiamate con argomento decrescente

```
int factRic(int n)
{
  if (n==0) return 1;
  else return factRic(n);
}
```

Catena infinita di chiamate identiche

Problemi nell'uso della ricorsione

- Uso della memoria
  - La programmazione ricorsiva comporta spesso un uso inefficiente della memoria per la gestione degli spazi di lavoro delle chiamate generate
  - In alcuni casi viene comunque preferita ad altri approcci per la sua eleganza e semplicità
  - In altri casi, si può ricorrere ad implementazioni iterative
  - Esempio

```
int fibList(int n){
  int f1=1,f2=1,tmp;
  for (int i=3; i<=n; i++)
  {
      tmp = f2;
      f2 = f1+f2;
      f1 = tmp;
  }
  return f2;
}
```

Funzione iterativa che calcola il numero n di fibonacci