Ricorsione

Funzioni ricorsive

- Una funzione è detta ricorsiva se chiama, direttamente o indirettamente, se stessa.
- In C tutte le funzioni possono essere usate ricorsivamente.

- Un esempio di funzione ricorsiva è la funzione ***fatt*** che calcola il fattoriale sugli interi non negativi

  `fatt(n)=n!`

  definita come:

$$\texttt{fatt(n)} = \begin{cases} \texttt{1} & \texttt{se n<=1} \\ \texttt{n*fatt(n-1)} & \texttt{se n>1} \end{cases}$$

- La ricorsione è una tecnica di programmazione che si rivela particolarmente efficace per esprimere sinteticamente operazioni su tipi di dati ricorsivi, cioè tipi di dati i cui domini possono essere definiti in maniera induttiva (liste, pile,code, alberi).

Funzione fattoriale definita in modo ricorsivo

La funcione C ricorsiva che realizza il fattoriale è:

```
int fatt (int n)
{ if(n<=1) return 1;
   else    return n*fatt(n-1);
}
```

- La stessa funzione C può essere realizzata in modo iterativo nel seguente modo:

```
int fatt (int n)
{ int i,f=1;
  for(i=n;i>=2;i--)
    f=f*i;
  return f;
}
```

Funzione somma dei primi n numeri naturali definita in modo ricorsivo.

- La somma dei primi n numeri naturali può essere definita per induzione come:

$$\text{Somma}(n) = \begin{cases} n & \text{se } n<=1 \\ n+\text{somma}(n-1) & \text{se } n>1 \end{cases}$$

- La funzione C che realizza in modo ricorsivo la somma dei primi n numeri naturali pertanto è:

```
int somma (int n)
{
   if (n<=1) return n;
   else  return n+somma(n-1);
}
```

- La versione iterativa della stessa funzione è:

```
int somma (int n)
{ int som=0;

  for(i=n;i>0;i--)
     som+=i;
  return sum;
}
```

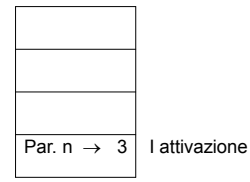
Modello di attivazione delle funzioni definite in modo induttivo.

- Durante l'attivazione di una funzione ricorsiva, i parametri formali e le eventuali variabili locali vengono inserite sulla cima dello stack.
- Ad esempio nel calcolo del fatt(3)

I attivazione (fatt(3))

– Viene inserito in cima allo stack il parametro n in cui viene copiato 3;

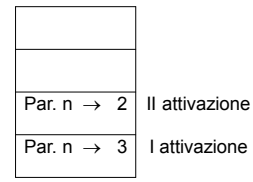
– Poiché n>1, viene invocato fatt(2);



II attivazione (fatt(2))

– Viene inserito in cima allo stack il parametro n in cui viene copiato 2;

– Poiché n>1, viene invocato fatt(1);

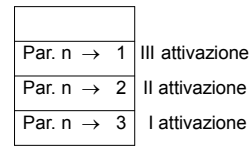


Modello di attivazione delle funzioni definite in modo induttivo.

III attivazione (fatt(1))

– Viene inserito in cima allo stack il parametro n in cui viene copiato 1;

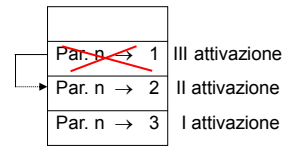
– Poiché n==1, la funzione fatt non viene ulteriormente invocata;



Fine III attivazione

– Viene restituito il valore 1;

– Viene eliminato dallo stack il parametro formale n.

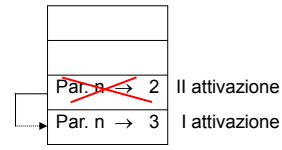


Fine II attivazione

– Il valore restituito dalla III attivazione (1) viene moltiplicato a 2;

– Viene restituito il risultato del prodotto (2*1=2);

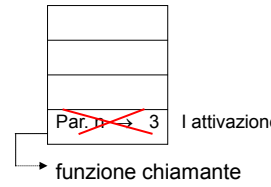
– Viene eliminato dallo stack il parametro formale n.



Modello di attivazione delle funzioni definite in modo induttivo.

Fine I attivazione

- Il Valore restituito dalla II attivazione (2) viene moltiplicato a 3;
- Viene restituito il risultato del prodotto (3*2=6);
- Viene eliminato dallo stack il parametro formale n.



•L'esecuzione della funzione fatt(n) ha richiesto di inserire n volte nello stack il parametro formale, di restituire n volte il valore di ogni attivazione e liberare n volte dallo stack il parametro formale.

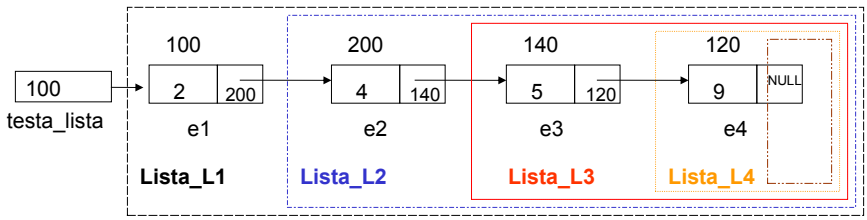
•Poiché tutte queste attività richiedono del tempo, la versioni ricorsive delle funzioni sono spesso meno efficienti delle analoghe versioni iterative in cui è richiesta una sola volta l'allocazione dei parametri e la successiva liberazione al termine della funzioni.

Modello di attivazione delle funzioni definite in modo iterativo

- Durante l'attivazione della versione iterativa della funzione fatt:
  - Vengono allocate in cima allo stack il parametro formale n e le variabili locali i, f;
  - Viene calcolato il fattoriale;
  - Viene restituito il valore del fattoriale;
  - Vengono eliminate dalla cima dello stack le variabili precedentemente allocate.

Funzioni ricorsive per una lista

- Una lista può essere definita in modo induttivo nel seguente modo:
  - Una lista è una sequenza vuota di elementi oppure una sequenza formata da un elemento appartenente al dominio atomo seguito da un valore di tipo lista.
  - Es. L=() oppure L= (e, L1) dove L1 è una lista.
- Tale definizione si rivela particolarmente utile per esprimere algoritmi ricorsivi sulle liste.



Lista_L1=(e1, Lista_L2=(e2, Lista_L3=(e3, Lista_L4=(e4, ()))))

Visualizzazione degli elementi di una lista in modo ricorsivo

Data una lista di interi definita come:

```
struct atomo { int elemento;
               struct atomo *next;
             };
struct atomo *testa_lista;
```

Per la definizione induttiva della lista data in precedenza, la visualizzazione in modo ricorsivo è definita nel seguente modo:

- se la lista è vuota, non visualizza alcun valore;
- se la lista è non vuota:
  - Visualizza l'elemento in testa;
  - Visualizza la lista che segue l'elemento in testa;

La versione C della funzione di visualizzazione ricorsiva è la seguente:

```
void visualizza (struct atomo *tl)
{ if (tl)
      { printf("%d",tl->elemento);
                visualizza(tl->next);
        }
}
```

Inserimento ordinato in una lista in modo ricorsivo

- Nell'inserimento ordinato ricorsivo di un elemento **el** in una lista si considerano i seguenti casi:
  - *la lista è vuota*: viene fatto un inserimento in testa alla lista;
  - *l'elemento **el** da inserire è minore dell'elemento in testa*: viene fatto un inserimento in testa alla lista;
  - *l'elemento **el** da inserire è maggiore dell'elemento in testa*: viene fatto un inserimento ordinato nella lista che segue l'elemento in testa.
- Per realizzare l'ultimo caso è necessario invocare in modo ricorsivo la funzione di inserimento ordinato, passando per indirizzo il puntatore all'elemento successivo al primo.

- La versione C ricorsiva della funzione di inserimento ordinato è:

```
void inserimento_ord(struct punt **ptl, int el)
{
  if(lista_vuota(*ptl) || el < (*ptl)->elemento )
        in_testa(ptl,el);
  else  inserimento_ord (&(*ptl)->next,el);
}
```

`(*ptl)->next` è il puntatore all'elemento successivo al primo.

Cancellazione da una lista in modo ricorsivo

- Nella cancellazione ricorsiva di un elemento **el** da una lista si considerano i seguenti casi:
  - *la lista è vuota*: non viene fatta alcuna attività;
  - *l'elemento **el** da cancellare è in testa alla lista*: viene fatta una cancellazione dalla testa;
  - *l'elemento **el** da inserire non è in testa*: viene fatta una cancellazione dell'elemento **el** nella lista che segue l'elemento in testa.
- Per realizzare l'ultimo caso è necessario invocare in modo ricorsivo la funzione di cancellazione, passando per indirizzo il puntatore all'elemento successivo al primo.

- La versione C ricorsiva della funzione cancella è la seguente:

```
void cancella( struct punt **ptl, int el)
{
  if (lista_vuota(*ptl)) return ;
  if((*ptl)->elemento==el)
       da_testa(ptl);
  else cancella(&(*ptl)->next,el);
}
```