

Laboratorio di Algoritmi e Strutture Dati

Esercitazioni del 27 Novembre 2012

Esercizio 1: Eliminazione della ricorsione in coda

Una funzione f è *ricorsiva in coda* se, per ogni possibile esecuzione, l'eventuale chiamata ricorsiva a f è l'ultima istruzione ad essere eseguita.

Ecco uno schema abbastanza generale per una funzione ricorsiva in coda:

```
/* ordine d'esecuzione: v,condition,g; v,condition,g; ...; v,condition,g; h,i. */
int recf(int x, double y, ...)
{
    int v = v0;
    char c = c0;
    ... = ...;

    if(condition(x,y,...)) {
        g(...);
        /* calcolo di x1, y1, ... */
        return recf(x1,y1,...);
    }
    h(...);
    return i(...);
}
```

La ricorsione può sempre essere eliminata: vale a dire, si può sempre riscrivere la funzione in modo da sostituire le chiamate ricorsive con costrutti iterativi. In particolare, è molto semplice eliminare la ricorsione in coda.

Ecco come riscrivere iterativamente la funzione ricorsiva in coda:

```
/* ordine d'esecuzione: v,condition,g; v,condition,g; ...; v,condition,g; h,i. */
int iterf(int x, double y, ...)
{
    int v = v0;
    char c = c0;
    ... = ...;

    while(condition(x,y,...)) {
        g(...);
        /* calcolo di x1, y1, ... */
        x = x1;
        y = y1;
        ... = ...;
        v = v0; /* ri-inizializzazione di v */
        c = c0; /* ri-inizializzazione di c */
        ... = ...;
    }
    h(...);
}
```

```

    return i(...);
}

```

(il tipo dei parametri e delle variabili locali di `recf` è specificato solo a scopo esemplificativo, la tecnica per l'eliminazione della ricorsione in coda si può applicare per ogni scelta di tipi.)

Esercizio:

Si consideri la seguente funzione che implementa tramite ricorsione in coda la ricerca binaria di un elemento n nel sottoarray $a[l], a[l+1], \dots, a[r-1]$ di un array a di interi.

```

/* a e' un array di k interi ordinato in modo crescente */
/* a[0] <= a[1] <= ... <= a[k-1] */
/* l e r sono due indici 0 <= l <= r < k */
/* ibinsearch restituisce l'indice m, l <= m < r di un elemento di valore n */
/* oppure -1 se tale elemento non e' contenuto in {a[l],a[l+1],...,a[r-1]} */

int rbinsearch(int *a, int l, int r, int n)
{
    int m = (l+r)/2;

    if(a[m] == n)
        return m;
    if(m == l)
        return -1;
    if(a[m] > n)
        return rbinsearch(a,l,m,n);
    else
        return rbinsearch(a,m+1,r,n);
}

```

Si riscriva `ibinsearch` iterativamente, tramite l'eliminazione della ricorsione in coda.