Compito di Programmazione

28 giugno 2010

Teoria

(1) Dato l'array char X[3][6][10][20], rispondere ai seguenti due punti:
(i) che tipo ha *(*X-4)-4 e che differenza c'è tra il suo valore e quello di X?
(ii) Che tipo ha X[0] e che differenza c'è tra il suo valore e quello di X?

(2) Il seguente programma è corretto o no? Spiegate la vostra risposta e, se pensate sia corretto, spiegate cosa stampa e perché:

```
int & g(int ** x){int *p=*x+1; *p=**x+*p; return *(p-1);}
main() {int X[]={1,2,3}, *q=X+1; g(&q)=X[1]; cout<<X[0]<<X[1]<<X[2]<<endl;}
```

Problema: abbiamo un intero m ed un array Y di n interi e vogliamo determinare se esiste un insieme di indici i1,…ik in [0,n-1] tale che Y[i1]+Y[i2]+…+Y[ik]=m. Chiamiamo un tale insieme di indici i1,…ik una **soluzione per m in Y** ed una rappresentazione di questa soluzione è un array R di n booleani tale che, per ogni j in [0,n-1], R[j]=true se j è nella soluzione i1,…ik e altrimenti è false. **Non si fa nessuna ipotesi sull'intero m e sugli interi contenuti in Y.**

Programmazione iterativa: scrivere una funzione iterativa bool F(int m, int* Y, int n, bool* R) che restituisca true se e solo se esiste una soluzione per m in Y e in questo caso R deve essere la rappresentazione della soluzione trovata. Se invece non ci sono soluzioni per m in Y, allora F deve restituire false (e valori qualsiasi in R).

a) Scrivere la PRE e POST condizione della funzione F;

b) realizzare la funzione F assumendo di avere a disposizione la funzione bool add_one(bool *R, int n) che soddisfa la seguente coppia di PRE e POST.

PRE=(n >0 e R[0..n-1] è definito, e chiamiamo VAL(R) il valore rappresentato da R[0..n-1] interpretato come numero binario (con true=1 e false=0)) ; vedi esempio più sotto.

POST=(se VAL(R)+1 è rappresentabile come binario con n bit (cioè VAL(R)+1 < 2^n) allora add_one restituisce false e modifica R in modo che rappresenti in binario VAL(R)+1, se invece VAL(R)+1 non è rappresentabile con n bit, allora add_one restituisce true)

Esempio *sia n=3 e R=[true, false, false], allora VAL(R)=4. Se invochiamo add_one con questo R e n=3, la funzione deve restituire false e R=[true,false,true] il cui VAL è 4+1. Se invece n=3 e R=[true,true,true], VAL(R)= 7 e visto che 7+1 non è rappresentabile con 3 bit, add_one deve restituire true (e qualsiasi valore per R). Intuitivamente, il valore booleano restituito da add_one ci dice se sommando 1 ad R causiamo overflow (true) oppure no (false).*

c) Scrivere l'invariante del ciclo principale della vostra funzione F.

Programmazione ricorsiva: scrivere una funzione ricorsiva bool F_RIC(int m, int* Y, int n, bool *R) che obbedisca alle stesse specifiche della parte iterativa.

a) Scrivere la PRE e POST condizione della funzione F_RIC;

b) Realizzare la funzione ricorsiva F_RIC. Questa funzione deve seguire lo schema classico delle funzioni ricorsive. **Non deve seguire l'idea suggerita per l'esercizio iterativo.**

c) Dimostrare per induzione che la funzione F_RIC è corretta rispetto alla PRE e POST del punto (a).