A

Ingegneria degli Algoritmi (A.A. 2012-2013)

Corsi di Laurea in Ingegneria Informatica e Automatica, Ingegneria dei Sistemi Informatici, e Laurea Magistrale in Ingegneria Informatica

Sapienza Università di Roma

Appello del 17/06/2013 – Compito A – Durata 45'

Domanda 1 (2 punti)

Si consideri la seguente funzione C, che calcola ricorsivamente la somma degli elementi di una lista collegata:

```
void sum(node* p, int* s) {
    if (p==NULL) return;
    *s += p->elem;
    sum(p->next, s);
}
```

dove `node` è una struttura con un campo `elem` di tipo `int` e un campo `next` di tipo `node*`, che punta al nodo successivo nella lista. Riformulare opportunamente la funzione in modo da eliminare la ricorsione di coda.

Domanda 2 (2 punti)

Si traduca in italiano la seguente dichiarazione C:

```
int (*v(int (*x)(int)))[10];
```

Domanda 3 (2 punti)

A cosa servono i boundary tag in un allocatore di memoria? Ove un allocatore ne faccia uso, è necessario che siano presenti in tutti i blocchi? Motivare le risposte.

Domanda 4 (2 punti)

Il codice x86-64 a sinistra non è equivalente alla funzione C a destra[1]. Perché?

```
f:  pushq %rbp
    movq  %rsp, %rbp
    movq  (%rsi,%rdx,8), %rax
    movq  %rax, (%rdi,%rdx,8)
    popq  %rbp
    ret
```

```
void f(int* a, int* b, long i) {
    a[i] = b[i];
}
```

1. Modificare il codice x86-64 in modo che sia equivalente al codice C a destra.
2. Fornire almeno due varianti diverse di funzione `f` che siano equivalenti al codice x86-64 a sinistra (nomi diversi degli identificatori non contano :-).

---

[1] Si ricordi che l'ordine dei primi tre parametri passati nel System V AMD64 ABI è rdi, rsi, rdx.