Fondamenti dell'Informatica
Prova in itinere

27 aprile 2016

Esercizio 1

Con la notazione $C^L_{L_1,L_2}$ indichiamo un compilatore da L_1 a L_2 scritto in L. Con $I^L_{L_1}$ indichiamo un interprete scritto in L per il linguaggio L_1.

Supponiamo che A sia il linguaggio della macchina M. Supponiamo che X sia un nuovo linguaggio che desideriamo implementare su M. Possiamo fare questo:

- scegliendo un opportuno $S \subseteq X$;
- creando un compilatore da S in A scritto in A;
- ... che altro?

Descrivere formalmente l'intero procedimento.

Avendo implementato X su M, come potremmo ottenere un'implementazione di X per una macchina M' il cui linguaggio è A'?

Esercizio 2

Nello scope delle dichiarazioni:

```
int n;
string s;
int g(int x, real y) {...}
```

si consideri l'espressione `g(f(n), f(s))`. Si diano ipotesi sul linguaggio e/o sul nome `f` affinché tale espressione sia corretta rispetto ai tipi.

Esercizio 3

Si consideri il seguente programma:

```
void swap (int a, int b) {
  int temp;
  temp = a;
  a = b;
  b = temp;
}

void main () {
  int value = 2, list[5] = {1, 3, 5, 7, 9};
  swap(value, list[0]);
  swap(list[0], list[1]);
  swap(value, list[value]);
}
```

Per ognuna delle seguenti modalità di passaggio dei parametri, si dica quali sono i valori delle variabili `value` e `list` dopo ognuna delle tre chiamate a `swap`: per valore, per riferimento, per nome, per valore-risultato.

Esercizio 4

Si assuma che in un generico linguaggio imperativo a blocchi, il blocco A contenga una chiamata della funzione f. Il numero dei record di attivazione (RdA) presenti a run-time sulla pila fra il RdA di A e quello della chiamata di f è fissato staticamente (ovvero è determinable a tempo di compilazione) o invece può variare dinamicamente (e dunque può essere determinato solo a tempo di esecuzione)? Motivare la risposta.

Esercizio 5

Si consideri la seguente definizione di tipo record:

```
type S = struct {
  int x;
  char y;
};
```

Si supponga che un `int` sia memorizzato su 2 byte, un `char` su 1 byte, su un'architettura a 16 bit con allineamento alla parola. In un blocco viene dichiarato un vettore:

```
S A [10];
```

Indicando con `PRDA` il puntatore all'RdA di tale blocco, e con `ofst` l'offset tra il valore di `PRDA` e l'indirizzo iniziale di memorizzazione di `A`, si dia l'espressione per il calcolo dell'indirizzo dell'elemento `A[i].y`.

Esercizio 6

Sono date le seguenti definizioni di funzione:

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

Una certa implementazione del linguaggio si comporta nel modo seguente. Alla chiamata `fact_falso(1)`, non risponde, rimanendo in un loop infinito. Alla chiamata `fact_vero(-1)` risponde dopo qualche tempo con `Stack overflow during evaluation`, abortendo l'esecuzione. Si dia una spiegazione motivata di questi due fatti.

Esercizio 7

Si spieghi la differenza tra iterazione determinata ed iterazione indeterminata. Si dica, motivando la risposta, se un linguaggio con allocazione statica della memoria può contenere un comando di iterazione indeterminata.

Esercizio 8

In un linguaggio di programmazione non meglio specificato, l'espressione `1 + 3.14` valuta all'intero 4, mentre l'espressione `3.14 + 1` valuta al numero razionale $4,14000034332275390625$. Si espongano deduzioni plausibili a partire da queste ipotesi.

Sotto le stesse ipotesi, si dica a cosa valutano le seguenti quattro espressioni: `1 + 1`, `1 + 0.75`, `0.75 + 1`, `0.75 + 0.75`.