Fondamenti dell'Informatica
Prova in itinere

26 aprile 2017

Esercizio 1

Quando emerse la necessità di un'implementazione del Pascal sulle macchine ICL, gli unici compilatori Pascal esistenti producevano codice CDC: $C^{\text{CDC}}_{\text{Pascal,CDC}}$ e $C^{\text{Pascal}}_{\text{Pascal,CDC}}$. Uno di questi compilatori fu modificato manualmente. Non furono scritti/modificati altri compilatori e non furono scritti interpreti. Si illustri formalmente un procedimento efficiente mediante il quale, in queste condizioni e avendo a disposizione una macchina CDC, si può ottenere un compilatore Pascal che gira su macchine ICL e produce codice ICL.

Esercizio 2

È dato il seguente programma in uno pseudo-linguaggio che ammette eccezioni:

```
void foo() throws Exc {
  throw new Exc(); }
void h(int X) throws Exc {
  if (X==1) { foo(); }
  try { foo(); } catch (Exc p) { **H1** } }
...
  int Y;  read(Y); try { h(Y); } catch (Exc p) { **H2** }
```

È possibile che venga eseguito `H2`? Se sì, sotto quali condizioni?

Esercizio 3

Si dica cosa stampa il seguente frammento in uno pseudolinguaggio con passaggio per nome e scope dinamico:

```
int x = 4;
void foo(name int y) {
  int x = 6; int w; x = x + y; w = y; write(x); write(y); }
{ int x = 10; foo (x++); write(x ); }
```

Esercizio 4

Si dica cosa stampa il seguente frammento in uno pseudolinguaggio con scope statico nei due casi passaggio per riferimento e passaggio per valore-risultato:

```
int x = 3; int y = 4;
void foo(mode int y, mode int z) {
  int x = 5; y = y +1; if (z == y) write(x); else write(y); }
foo(x, x); write(x); write(y);
```

Esercizio 5

Si consideri la seguente definizione di tipo record:

```
type S = struct {
  int x;
  char y;
  float z[20];
};
```

Si supponga che `int` e `float` siano memorizzati su 4 byte e `char` su 1 byte, su un'architettura a 32 bit con byte di 8 bit e con allineamento di `int` e `float` 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].z[j]`.

Esercizio 6

Si dica cosa viene stampato dal seguente frammento di codice, scritto in uno pseudolinguaggio che ammette iterazione determinata espressa con il costrutto for implementato mediante *iteration count*:

```
x = 1;
for i = 1 to 3+x by 1 do {
  write(i); x++;
}
write(x+1);
```

Esercizio 7

Si spieghi cosa si intende per "ricorsione di coda". Sia dato un tipo di dato `T` per la rappresentazione dei numeri naturali e le dichiarazioni:

```
bool is_zero(T x); // Vero se e solo se x = 0.
T pred(T x);       // Se x != 0, restituisce x - 1.
T add(T x, T y);   // Restituisce x + y.
```

Si scrivano due funzioni

```
T mul(T x, T y);   // Restituisce x * y.
T mulrc(T x, T y); // Restituisce x * y.
```

in cui la prima non usi ricorsione di coda e la seconda usi ricorsione di coda. Non è lecito fare alcuna ipotesi aggiuntiva su `T`, se non che le funzioni del primo gruppo fanno quanto promesso. Le funzioni del secondo gruppo non devono fare riferimento a tipi diversi da `T` e `bool` e non devono usare costrutti iterativi.

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 $4,140000334332275390625$. 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`.