Corso di Linguaggi di Programmazione

TERZO APPELLO A.A. 2006/2007

C. Braghin e A. Ceselli

3 Settembre 2007

Cognome e Nome: .....

Numero matricola: .....

Domanda 1. Si discuta brevemente il meccanismo di gestione esplicita delle *eccezioni* a livello di linguaggio presente nei moderni linguaggi di programmazione. In particolare, si considerino brevemente almeno i seguenti punti: (i) definizione di eccezione, (ii) scelte nella definizione della gestione delle eccezioni: chi le gestisce, chi le deve gestire, dove e come devono venire specificati i gestori di eccezioni, cosa succede se si solleva un'eccezione che non viene gestita da nessuna parte del codice (e.g., si tratta di un errore di compilazione o run-time)? (iii) errori sintattici e/o lessicali possono venire considerati eccezioni? E gli errori di tipo? Giustificare la risposta.

Domanda 2. Si presentino analogie e differenze tra il meccanismo di tipi e sottotipi ed il meccanismo dei tipi parametrici, evidenziando i reciproci vantaggi e svantaggi. Se ne discuta l'implementazione, anche attraverso esempi, in almeno un linguaggio di programmazione che incorpora entrambi i meccanismi.

Domanda 3. Scrivere in Scheme una funzione `rimuovi` che prende come argomento un simbolo e una lista e ritorna come risultato la lista ottenuta eliminando tutte le occorrenze del simbolo al suo interno. Ad esempio:

```
> (rimuovi 'a '(a a b c d) )
(b c d)
> (rimuovi 'a '(b c d) )
(b c d)
> (rimuovi ' a '(a (1 a) () (1 2 3)) )
((1) () (1 2 3))
> (rimuovi '1 '(1 2 3) )
(2 3)
```

Domanda 4. Assumendo le seguenti definizioni:

```
(define lista1 '((1 2) (3 4) (5 6)) )
(define lista2 '(a b c) )
(define lista3 '(100 200 300) )
```

```
(define a 0)
(define b 3)
(define c 6)
(define risultato (let ((a 1)
                        (b (* a 10))
                        (c (+ b a)))
(list a b c)) )
```

scrivere i valori restituiti dalla valutazione delle seguenti espressioni:

```
(cdar lista1)
(append lista3 lista1)
(map cdr lista1)
(map cons lista2 lista3)
risultato
```

Domanda 5. Realizzare un programma Prolog che, ricevute in ingresso due liste di interi ordinate, potenzialmente contenenti elementi duplicati, restituisce una lista ordinata degli interi contenuti nelle due liste, ma senza elementi duplicati. Ad esempio:

```
?- dunion([1,2,3,3],[2], R).
R = [1,2,3].
```

Che risultato si può attendere utilizzando il predicato nel seguente modo:

```
?- dunion(X,[2],[1,2,3]).
```

Perchè?