Corso di Laurea in Informatica Applicata
Primo Appello di Programmazione II
4 Febbraio 2011

Istruzioni: scrivere le risposte alle domande negli spazi presenti nel testo e riconsegnare solo questi fogli. Non è consentito l'uso di libri o appunti. Inserire il proprio nome e cognome, numero di matricola e firma in calce a tutti i fogli.

Tempo a disposizione: 2 ore.

Domanda 1 (4 punti)

Per ogni domanda, selezionare tutte e solo le risposte giuste.

(1 punto) (a) Cosa indica la notazione $I_{\mathcal{L}}^{\mathcal{L}o}$?

A. Un programma scritto in $\mathcal{L}o$ che sa interpretare tutte le istruzioni di $\mathcal{L}$.

B. Un programma scritto in $\mathcal{L}$ che sa interpretare tutte le istruzioni di $\mathcal{L}o$.

C. Nessuna delle precedenti.

(1 punto) (b) Quale dei seguenti tipi è un tipo scalare:

A. array.

B. char.

C. record.

D. enumerazioni.

E. intervalli

F. puntatori.

(1 punto) (c) La postcondizione EFFECTS:

A. deve essere verificata prima della chiamata.

B. indica ciò che vale alla fine della chiamata.

C. suppone che siano verificate le precondizioni.

D. assente se la procedura ha come tipo di ritorno `void`.

(1 punto) (d) Un'interfaccia in Java:

A. può avere variabili d'istanza.

B. può avere metodi concreti.

C. è possibile creare variabili riferimento di tale interfaccia.

D. non è possibile creare oggetti di tale interfaccia.

E. può estendere un'altra interfaccia con la parola chiave `extends`.

Nome e Cognome Matricola Firma

Domanda 2 (4 punti)

(2 punti) (a) Si dia la definizione di binding.

Risposta: *Un binding è un'associazione, come quella che avviene tra un attributo e un'entità o tra un operazione e un simbolo (operatore).*

(2 punti) (b) Si dica la differenza tra binding statico e binding dinamico.

Risposta: *Un'associazione è statica se avviene prima dell'esecuzione e resta immutata per tutta l'esecuzione del programma. Un'associazione è dinamica se avviene per la prima volta durante l'esecuzione e può essere cambiata nel corso dell'esecuzione del programma.*

Domanda 3 (8 punti)

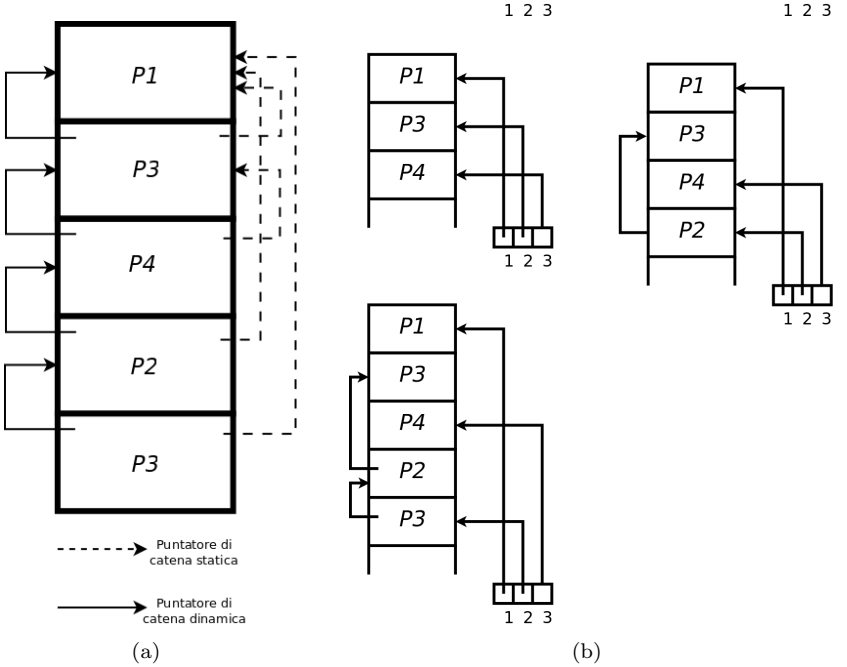
È dato il seguente frammento di codice in uno pseudo-linguaggio con scope statico:

```
void P1 (){
   void P2 (){
      //corpo di P2
      }
   void P3 (){
      void P4 (){
         //corpo di P4
         }
      //corpo di P3
      }
   //corpo di P1
}
```

(4 punti) (a) Si descriva graficamente la pila dei record di attivazione (solo quella finale), limitatamente ai puntatori di catena statica e dinamica, dopo la successione di chiamate (tutte attive) `P1`, `P3`, `P4`, `P2`, `P3`.

(4 punti) (b) Si supponga che, invece della catena statica, si utilizzi la tecnica del display. Si descriva graficamente (passo dopo passo) la pila e il display dopo la successione di chiamate (tutte attive) `P1`, `P3`, `P4`, `P2`, `P3`.

Risposta: *Situazione analoga a invocare `P1()` che nel suo corpo invoca `P3()` che nel suo corpo invoca `P4()` etc.*



Domanda 4 (6 punti)

In uno pseudo-linguaggio con eccezioni, si incontra il seguente frammento di codice:

```
1  public static void ecc() throws X
2  {
3     throw new X();
4  }
5
6  public static void g(int para)
7  {
8     if(para == 0) ecc();
9     try { ecc(); } catch(X) { write(3); }
10 }
11
12 public static void main
13 {
14    try { g(1); } catch(X) { write(1); }
15    try { g(0); } catch(X) { write(0); }
16 }
```

Indicare cosa viene stampato, giustificando la risposta.

Risposta: *Quando il metodo* `g` *è invocato con il parametro avente valore 1 (riga 14), la guardia dell'*`if` *a riga 8 risulta falsa e quindi si esegue il costrutto* `try-catch` *alla riga successiva: all'interno di questo, viene invocato il metodo* `ecc` *che lancia l'eccezione* `X` *(riga 3). Essendo questa catturata all'interno del metodo* `g` *nel costrutto* `try-catch` *che contiene l'invocazione a* `ecc`*, viene eseguito il gestore relativo che stampa il valore 3. Se il metodo* `g` *è invocato con il parametro avente valore 0 (riga 15), la guardia dell'*`if` *a riga 8 risulta vera e quindi viene invocato il metodo* `ecc` *che lancia l'eccezione* `X` *(riga 3). Questa eccezione non è gestita dal metodo* `g` *che quindi la trasmette al chiamante (in questo caso, il* `main`*). Dato che il chiamante ha effettuato l'invocazione a* `g` *all'interno di un costrutto* `try-catch` *che gestisce proprio il tipo dell'eccezione sollevata, viene eseguito il gestore stesso che stampa il valore 0.*

Domanda 5 (10 punti)

Data la seguente parziale implementazione di una classe in Java:

```
1 public class Employee implements Comparable<Employee> {
2    private String name;
3    private double salary;
4    ...
5 }
```

Si fornisca l'implementazione:

(2 punti) (a) del costruttore con due parametri, per inizializzare le due variabili d'istanza;

(4 punti) (b) di due metodi accessori, per ritornare il valore delle variabili d'istanza;

(4 punti) (c) del metodo di firma `public int compareTo(Employee other)` per confrontare un `Employee` con un altro in base a `salary`. Questo metodo deve ritornare un numero negativo, zero o un numero positivo se, rispettivamente, l'oggetto su cui viene invocato è minore, uguale o maggiore dell'oggetto passato come parametro.

```
public class Employee implements Comparable<Employee>
{
   String name;
   double salary;

   public Employee(String name, double salary)
   {
      this.name = name;
      this.salary = salary;
   }

   public String getName()
   {
      return name;
   }

   public double getSalary()
   {
      return salary;
   }

   public int compareTo(Employee other)
   {
      if(salary < other.salary) return -1;
      if(salary > other.salary) return 1;
      return 0;
   }
}
```