CC60H : Nuevos Lenguajes de Programación
Control 1–Semestre Otoño'2000
Prof.: Luis Mateu.

Sin apuntes, 1 hora 20 minutos

Pregunta 1

El siguiente programa contiene líneas con errores :

```
Map m= new HashMap();
m.put("gato", "mamifero felido domestico");
m.put("leon", "mortifero carnicero felido");
m.put("pi", 3.14);
String s= m.get((Object)"gato"));
String s2= m.get("leon");
double pi= (double)m.get("pi");
m.put("pi", "3.14");
m.remove("gato");
```

a.- Explique por qué esas líneas están incorrectas.

b.- Corrija las líneas que contienen errores (sólo las que tienen errores y respetando la intención original del programador).

c.- Indique cuál será el contenido final del arreglo asociativo después de ejecutar el programa.

Pregunta 2

La siguiente figura muestra a la izquierda declaraciones de variables e invocaciones de métodos. A la derecha se definen las clases A, B y C. Observe que B y C son subclases de A.

```
A aa= new A();
B  bb= new B();
C cc= new C();
A ba= bb;
A ca= cc;

ba.m(bb, aa);
ba.m(ba, aa);
ba.m(aa, aa);
ca.m(aa, aa);
aa.m(cc, bb); // Ojo!
```

```
class A {
  void m(A a1, A a2) { System.out.println( "A.m(A,A)" ); }
  void m(B b, A a) { System.out.println( "A.m(B,A)" ); }
  void m(A a, C c) { System.out.println( "A.m(A,C)" );}
}

class B extends A {
  void m(A a1, A a2) { System.out.println( "B.m(A,A)" ); }
}

class C extends A {
  void m(A a, C c) { System.out.println( "C.m(A,C)" ); }
}
```

Indique en forma precisa qué despliegan en pantalla cada una de las invocaciones de métodos.

Pregunta 3

En la siguiente expresión, m es de tipo Map y a es de tipo Alumno (ver pregunta 4):

```
!a.ci.equals("") ? ((Double)m.get(a.ci)).doubleValue() : 1
```

Construya para esta expresión el árbol etiquetado con los tipos de cada subexpresión (considere que `.ci` es un operador unario).

Pregunta 4 (coef. 2)

La empresa Rivoc ha donado 100 buzos a la Facultad y el decano ha decido entregar estos buzos a los 100 mechones que hayan obtenido las mejores notas en el control 3 de Computación I.

Para hacer la repartición se dispone de un ArrayList (desordenado) con todos los alumnos de la Facultad. Todos los objetos en esta lista pertenecen a la clase Alumno en donde se puede obtener el carnet de identidad (String ci) y el nombre del alumno (String nom). Además se dispone de un segundo ArrayList (desordenado) con las notas de todos los alumnos que dieron el control 3. Cada objeto en esta lista pertenece a la clase Nota y contiene el carnet de identidad de un alumno (String ci) y su nota en el control (double nota), pero no aparece su nombre.

Utilizando las colecciones de Java, escriba un programa que muestre en pantalla los nombres de los 100 alumnos acreedores de un buzo *en orden alfabético y sin incluir la nota.* Para ello, ordene la lista de notas descendentemente por nota, luego construya un arreglo asociativo que asocie carnet de identidad con nombre de alumno, entonces construya un ArrayList con los nombres de los 100 alumnos con las mejores notas, ordénelos alfabéticamete y despliéguelos en pantalla.

Observación: Esta pregunta busca medir el buen uso de las colecciones. El mal uso será considerado incorrecto. Ud. debe priorizar la eficiencia en tiempo de programación, y no en tiempo de ejecución.

Torpedo:

```
interface Iterator {
  boolean hasNext();
  Object next();
}

interface List {
  boolean add(Object val);
  boolean add(int idx, Object val);
  Object get(int idx);
  Object set(int idx, Object val);
  Object remove(int idx);
  int size();
  Iterator iterator();
  ...
}

interface Map {
  Object get(Object key);
  Object put(Object key, Object val);
  int size();
  ...
}
```

```
interface Comparator {
  int compare(Object o1, Object o2);
}

class Collections {
  static void sort(List l);
  static void sort(List l, Comparator c);
  static void shuffle(List l);
  ...
}

class Integer {
  Integer(int i); // constructor
  int intValue();
  ...
}

class Double {
  Double(double x); // constructor
  double doubleValue();
  ...
}
```