9.D. Listas.

4. Listas (IV).

A la hora de usar las listas, hay que tener en cuenta un par de detalles, ¿sabes cuáles? Es sencillo, pero importante.

No es lo mismo usar las colecciones (listas y conjuntos) con objetos inmutables (Strings, Integer, etc.) que con objetos mutables. Los objetos inmutables no pueden ser modificados después de su creación, por lo que cuando se incorporan a la lista, a través de los métodos add, se pasan por copia (es decir, se realiza una copia de los mismos). En cambio los objetos mutables (como las clases que tú puedes crear), no se copian, y eso puede producir efectos no deseados.

Imaginate la siguiente clase, que contiene un número:

class Test
public Integer num;
<pre>Test (int num) { this.num=new Integer(num); }</pre>
La clase de antes es mutable, por lo que no se pasa por copia a la lista. Ahora imagina el siguiente código en el que se crea una lista que usa este tipo de objeto, y en el que se insertan dos objetos:
Test p1=new Test(11); // Se crea un objeto Test donde el entero que contiene vale 11.
Test p2=new Test(12); // Se crea otro objeto Test donde el entero que contiene vale 12.
LinkedList <test> lista=new LinkedList<test>(); // Creamos una lista enlazada para objetos tipo Test.</test></test>
THREULISCUEST IISta-New LINGUISCUEST/(), // Creamos una IIsta enlazada para objetos tipo rest.
lista.add(p1); // Añadimos el primero objeto test.
lista.add(p2); // Añadimos el segundo objeto test.
for (Test p:lista) System.out.println(p.num); // Mostramos la lista de objetos.
<pre>for (Test p:lista) System.out.println(p.num); // Mostramos la lista de objetos.</pre>
for (Test p:lista) System.out.println(p.num); // Mostramos la lista de objetos. ¿Qué mostraría por pantalla el código anterior? Simplemente mostraría los números 11 y 12. Ahora bien, ¿qué pasa si modificamos el valor de uno de los números de los objetos test? ¿Qué se mostrará al ejecutar el siguiente código?
¿Qué mostraría por pantalla el código anterior? Simplemente mostraría los números 11 y 12. Ahora bien, ¿qué pasa si modificamos el valor de
¿Qué mostraría por pantalla el código anterior? Simplemente mostraría los números 11 y 12. Ahora bien, ¿qué pasa si modificamos el valor de uno de los números de los objetos test? ¿Qué se mostrará al ejecutar el siguiente código?
¿Qué mostraría por pantalla el código anterior? Simplemente mostraría los números 11 y 12. Ahora bien, ¿qué pasa si modificamos el valor de uno de los números de los objetos test? ¿Qué se mostrará al ejecutar el siguiente código? p1.num=44; for (Test p:lista) System.out.println(p.num);
¿Qué mostraría por pantalla el código anterior? Simplemente mostraría los números 11 y 12. Ahora bien, ¿qué pasa si modificamos el valor de uno de los números de los objetos test? ¿Qué se mostrará al ejecutar el siguiente código? p1.num=44; for (Test p:lista) System.out.println(p.num); El resultado de ejecutar el código anterior es que se muestran los números 44 y 12. El número ha sido modificado y no hemos tenido que volver a insertar el elemento en la lista para que en la lista se cambie también. Esto es porque en la lista no se almacena una copia del objeto
¿Qué mostraría por pantalla el código anterior? Simplemente mostraría los números 11 y 12. Ahora bien, ¿qué pasa si modificamos el valor de uno de los números de los objetos test? ¿Qué se mostrará al ejecutar el siguiente código? pl.num=44; for (Test p:lista) System.out.println(p.num); El resultado de ejecutar el código anterior es que se muestran los números 44 y 12. El número ha sido modificado y no hemos tenido que
¿Qué mostraría por pantalla el código anterior? Simplemente mostraría los números 11 y 12. Ahora bien, ¿qué pasa si modificamos el valor de uno de los números de los objetos test? ¿Qué se mostrará al ejecutar el siguiente código? p1.num=44; for (Test p:lista) System.out.println(p.num); El resultado de ejecutar el código anterior es que se muestran los números 44 y 12. El número ha sido modificado y no hemos tenido que volver a insertar el elemento en la lista para que en la lista se cambie también. Esto es porque en la lista no se almacena una copia del objeto
¿Qué mostraría por pantalla el código anterior? Simplemente mostraría los números 11 y 12. Ahora bien, ¿qué pasa si modificamos el valor de uno de los números de los objetos test? ¿Qué se mostrará al ejecutar el siguiente código? p1.num=44; for (Test p:lista) System.out.println(p.num); El resultado de ejecutar el código anterior es que se muestran los números 44 y 12. El número ha sido modificado y no hemos tenido que volver a insertar el elemento en la lista para que en la lista se cambie también. Esto es porque en la lista no se almacena una copia del objeto Test, sino un apuntador a dicho objeto (solo hay una copia del objeto a la que se hace referencia desde distintos lugares).



O Falso.