Programación orientada a objetos. Examen febrero 2000.

Viernes 18 de febrero del 2000.

1. Definan **abstracción**, **encapsulación**, **jerarquía**, **modularidad**. ¿Cómo pueden estos elementos proveer la ilusión de simplicidad?
2. ¿Qué relación existe entre los conceptos **tipo**, **operación**, **mensaje**, **encapsulación**, **responsabilidad**? ¿Están presentes estos conceptos en Smalltalk? En tal caso ¿dónde?
3. Expliquen la implementación del siguiente método en la clase abstracta **`Collection`**. ¿En qué subclases de **`Collection`** puede funcionar y en qué clases es necesario sobreescribirlo?

```
select: aBlock

   |answer|

   answer := self class new.
   self do: [:element | (aBlock value: element)
      ifTrue: [answer add: element]].
   ^answer
```

4. En un lenguaje orientado a objetos en el que se define el tipo de las variables y argumentos ¿puede una variable o argumento declarado de una clase contener instancias de una clase ancestra? ¿Y de una sucesora? Justifiquen sus respuestas.
5. Implementar un método **`fibonnacci`** en la clase **`Integer`** para calcular el valor de la secuencia Fibonnacci que se define recursivamente así: para un número natural *n*:

   $fibonnacci(0) \equiv fibonnacci(1) \equiv 1$
   $fibonnacci(n + 1) \equiv fibonnacci(n) + fibonnacci(n - 1) \quad \forall n > 1$