Relación de problemas 1 (PDOO) **Curso: 2019-2020**

Conceptos de OO

1. Plantéate la siguientes cuestiones: ¿Qué es un paradigma? ¿Qué es un paradigma de programación? ¿Qué paradigmas de programación existen?

   Haz una lista con los distintos paradigmas de programación que encuentres, describiendo con tus propias palabras sus características principales.

   ¿Qué paradigma de programación has usado normalmente cuando has programado?

2. Responde a las siguientes cuestiones, no te conformes con lo expuesto en los primeros temas e investiga usando otras fuentes bibliográficas.
   a) En POO se suele decir: ***Todo es un objeto o un mensaje a un objeto,*** ¿se cumple esto en todos los lenguajes de programación? En caso negativo, cita al menos uno y explica por qué no lo cumple.
   b) Teniendo en cuenta los conocimientos que has adquirido previamente en otras asignaturas de programación, ¿cómo relacionarías los siguientes conceptos? clase, objeto, estado, método y envío de mensaje
   c) ¿Qué diferencia hay entre un tipo de dato y un tipo abstracto de dato (TAD)? y ¿entre un TAD y una clase? ¿Qué conceptos son específicos de la POO (no están presentes en el TAD)?
   d) Pon un ejemplo que muestre el concepto de herencia de forma parecida a como se ha hecho en las diapositivas de los primeros temas. Usa ese mismo ejemplo para explicar el concepto de polimorfismo.
   e) ¿Qué entiendes por tiempo de ligadura? Si los tipos de ligadura que existen según el tiempo en el que se llevan a cabo son estática y dinámica ¿cuál crees que has usado cuando has programado? ¿con qué lenguaje?
   f) ¿Qué problemas puede presentar un sistema software con bajo ocultamiento de la información?
   g) ¿Qué ventajas crees que aporta la OO en el diseño de software?

Lenguajes y técnicas de OO

1. Compara las características de Java y de Ruby con los lenguajes con los que hayas trabajo anteriormente (al menos con C++).
2. Realiza un estudio del uso y popularidad de los lenguajes de programación en los últimos años, para ello puedes visitar webs como el índice TIOBE, PYPL, el ranking RedMonk, el Programming Language Popularity Chart, o el Top Ten Programming Languages de IEEE. Añade algún otro ranking interesante a esta lista e intenta explicar a qué se deben las diferencias en la posición que ocupan los lenguajes en diferentes rankings.
3. Investiga sobre la historia y características de UML y explica para qué sirven y cómo

aparecieron los distintos tipos de diagramas, cuál es su uso en la actualidad, las ventajas e inconvenientes que tiene su uso.

4. ¿Existen detractores de UML? ¿cuál su opinión? ¿y la tuya?
5. Entra en los tutoriales oficiales y especificaciones de Ruby (https://ruby-doc.org/), Java (https://docs.oracle.com/javase/7/docs/api/index.html) y los que facilita Codecademy (https://www.codecademy.com/catalog/subject/all).

Reflexión sobre las fuentes

Presta atención a las fuentes de información que has manejado para resolver los ejercicios anteriores:

1. ¿Son fiables? ¿por qué? Si no lo son, consulta otras que sí lo sean y añade por qué son mejores que las que habías consultado antes.
2. ¿Las has citado convenientemente? ¿por qué? Si no las habías citado convenientemente hazlo. Ten en cuenta todos los aspectos relevantes (formato de la cita, respeto a la autoría, diferenciación de opinión/texto del autor del tuyo propio…).
3. ¿Son suficientes? ¿cubren todos los aspectos relevantes? ¿por qué?