UF2. [PAC01] Solución

Para responder a las siguientes cuestiones deberéis ayudaros del libro de texto, capítulo 4, y consultar, si lo creéis necesario, internet.

Debéis subir un único archivo comprimido que contenga UN ÚNICO DOCUMENTO PDF con las respuestas de la parte teórica.

Todas las preguntas valen lo mismo, 1 puntos cada una.

Recordad que la fecha límite para la entrega de esta PAC es el día 25 de NOVIEMBRE.

Información: Cualquier PAC copiada y/o en la que se haya utilizado "copy-paste" de código ya escrito será puntuada con un 0.

Actividades

Parte teórica

1. ¿Qué es el control de versiones?

La capacidad de recordar los cambios realizados tanto en la estructura de directorios como en el contenido de los archivos.

2. ¿Qué es subversión? ¿Cómo funciona el ciclo de vida en subversión?

Es una herramienta multiplataforma de código abierto para el control de versiones.

3. En control de versiones qué significa:

3.1. Repositorio.

Lugar donde se almacenan todos los datos y los cambios.

3.2. Revisión, versión.

Es una versión concreta de los datos almacenados.

3.3. Checkout.

Crear una copia de trabajo del proyecto, o de archivos y carpetas del repositorio en el equipo local.

3.4. Commit.

Se realiza commit cuando se confirman los cambios realizados en local para integrarlos al repositorio.

3.5. Import.

Es la subida carpetas y archivos del equipo local al repositorio.

3.6. Actualizar.

Se realiza una actualización cuando se desea integrar los cambios realizados en el repositorio en la copia de trabajo local.

4. ¿Qué es documentar el código de un programa?

Explicar claramente lo que hace el programa, de esta manera todo equipo de desarrollo sabrá lo que se está haciendo y por qué.

5. ¿Qué tipos de documentación se realizan en los proyectos?
   - Documentación de las especificaciones.
   - Documentación del diseño.
   - Documentación del código fuente.
   - Documentación de usuario final.

6. ¿Qué es JavaDoc? ¿Para qué se utiliza?

Es una utilidad de Java que se utiliza para extraer y generar documentación directamente del código en formato HTML.

7. ¿Qué es refactorización?

Es una técnica de la ingeniería de software que permite la optimización de un código previamente escrito.

- Limpia el código, mejorando la consistencia y la claridad.
- Mantiene el código, no corrige errores ni añade funciones nuevas.
- Va a permitir facilitar la realización de cambios en el código.
- Se obtiene un código limpio y altamente modularizado.

8. ¿Cuándo hay que refactorizar?
    - Cuando detectamos el mismo código en distintos lugares.
    - Cuando los métodos son muy largos.
    - Cuando una clase tiene muchos métodos, atributos e instancias.
    - Cuando se pasan muchos parámetros.
    - Cuando una clase es frecuentemente modificada.
    - Cuando se deben realizar cambios para compatibilizar con otros cambios en otras clases.
    - Cuando un método utiliza más elementos de otra clase que de la suya propia.
    - Cuando tenemos clases que sólo tienen atributos y métodos de acceso a ellos.
    - Cuando una subclase utiliza pocas características de sus superclases.

9. ¿Qué son los bad smells?

Son los síntomas para refactorizar:

- Código duplicado.
- Métodos muy largos.
- Clases muy grandes.
- Lista de parámetros extensa.
- Cambio divergente.
- Cirugía a tiro pistola.
- Clase de solo datos.
- Legado rechazado.

10. Cita métodos de refactorización en eclipse.
    - Rename
    - Move
    - Extract Constant
    - Extract Local Variable
    - Convert Local Variable to Field
    - Extract Method
    - Change Methos Signature
    - Inline
    - Member Type to Top Level
    - Extract Interface
    - Extract Superclass
    - Convert Anonymous Class to Nested