Hoja de trabajo No. 1

Realizar: Programa de Radio.
Realizarse: en **PAREJA.**

Objetivos:
a. Utilizar GIT (u otro software para control de versiones) para guardar las versiones del programa.
b. Practicar la definición de interfaces y reutilización de código.
c. Emplear JUnit para casos de prueba.
d. Repasar el diagrama UML de clases.
e. **Repasar el lenguaje JAVA**.

Programa a realizar:
Realizar el programa del ejercicio 1.13 de libro de Java Structures:

"Design a data structure to simulate the workings of a car radio. The state of the radio is on or off, and it may be used to listen to an AM or FM station. A dozen modifiable push buttons (identified by integers 1 through 12) allow the listener to store and recall AM or FM frequencies. AM frequencies can be represented by multiples of 10 in the range 530 to 1610. FM frequencies are found at multiples of 0.2 in the range 87.9 to 107.9."

Se debe tener un programa principal que emplee el radio implementado. Simplemente muestra un menú, o una interfaz GUI, con las opciones:

1. Prende el radio
2. Cambia de AM a FM a AM
3. Avanzar en el dial de las emisoras. Al llegar al final del dial inicia nuevamente.
4. Permite guardar una emisora en uno de los 12 botones
5. Permite seleccionar la emisora puesta en un botón
6. Apagar el radio

Tareas:
a. Diagrama UML de clases. Recuerde que la notación debe ser la correcta para las clases, interfaces y sus relaciones.
b. Construir la interface y la clase que implemente el radio de un carro.
c. Su programa principal debe permitir cambiar la clase que implementa la radio, sin que se vea afectado en su funcionamiento. Ver el requisito de calificación que se describe más adelante.
d. Debe dejar evidencia de todo el desarrollo en el repositorio de GIT (o de cualquier otro software para control de versiones). Indicar como acceder a su repositorio y si es necesario, agregar a su catedrático y auxiliar para que tengan acceso al mismo. Los correos de los auxiliares le serán proporcionados más adelante.

   Si se emplea GIT puede crear el repositorio en https://github.com/
   Haga el repositorio público para que sea gratuito.

   Su IDE le debe permitir trabajar con el control de versiones y repositorio seleccionado.
   En Eclipse puede consultar: http://wiki.eclipse.org/EGit/User_Guide#Basic_Tutorial:_Adding_a_project_to_version_control
   En Netbeans puede consultar: https://netbeans.org/kb/docs/ide/git.html

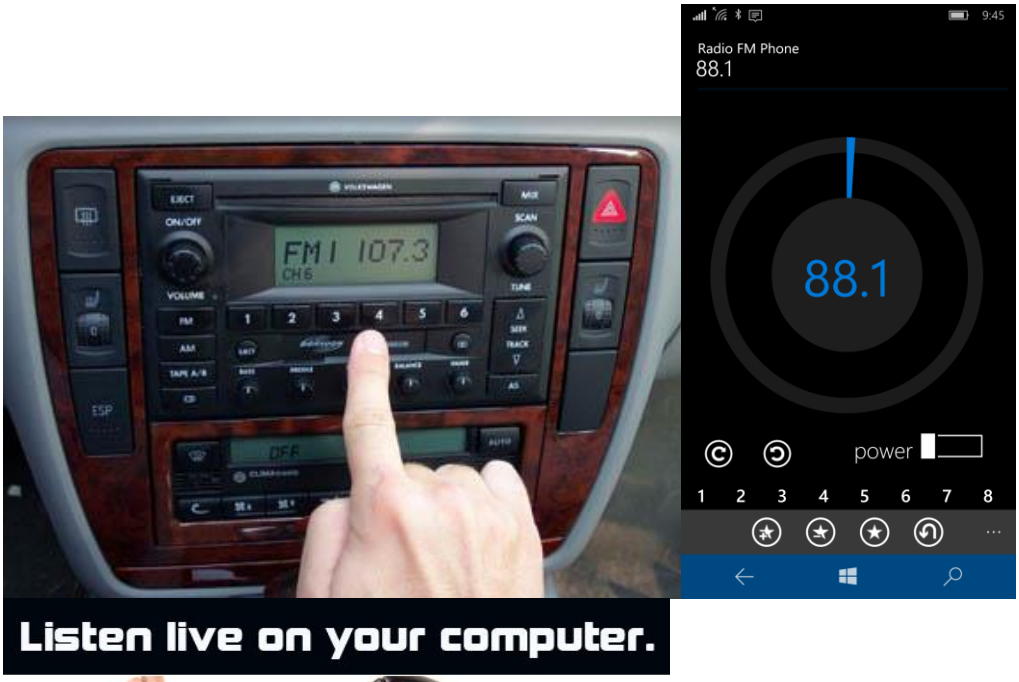
e. Incluya tres pruebas unitarias con JUnit. Use los test de versión 4.x y deben probar la funcionalidad de los métodos que usted desee. NO deben ser simplemente los tests generados automáticamente por el IDE, recuerde que tiene que modificarlos para que realmente prueben los métodos de su clase.

Su IDE (Netbeans o Eclipse) cuentan con la opción de JUnit, y solo debe configurarse o instalar un plugin. Por ello no es necesario instalar el framework completo.
Si usted emplea Netbeans puede consultar: https://netbeans.org/kb/docs/java/junit-intro.html
Un tutorial adicional (incluye como JUnit con Eclipse) es: http://www.vogella.com/tutorials/JUnit/article.html

Debe subir al Canvas todos los productos elaborados en los incisos a, b, c, e y los enlaces a su repositorio de GIT (o del sistema seleccionado para control de versiones). **Asegúrese de permitir ingreso a su repositorio a su catedrático y auxiliares del curso o que sea un repositorio público.**

Requisito para calificar esta hoja.
Su programa principal debe funcionar, **sustituyendo la clase que implementa la radio** por una clase producida por otros dos o tres grupos como mínimo. El cambio debe ser lo más trasparente posible en su programa principal, requiriendo que se **modifique una línea o menos** del mismo al momento de compilarse.

NOTA: para ejecutar este programa NO SE USARÁ ningún IDE. Debe poder iniciar su ejecución desde la línea de comandos del sistema operativo. No puede suponer que todos los desarrolladores emplearán un IDE específico, ni que se cuenta con clases que sean generadas automáticamente por el IDE.



Calificación: su programa debe funcionar y tener control de versiones para ser calificado.

Aspecto	Puntos
Estilo de codificación: comentarios, identación, nombres de variables significativas. Consultar "Code Conventions for the Java Programming Language"[1] o Google Java Style[2]	10
Documentación generada con Javadoc	10
Uso del repositorio: existen más de tres versiones guardadas, la última versión es igual a la colocada en el Blackboard. En el repositorio se encuentra evidencia que existen contribuciones en el código u otros documentos, realizadas por cada integrante del grupo.	10
Diagrama de clases: muestran la abstracción y encapsulación de las operaciones. Muestra correctamente las relaciones entre las clases. Se encuentra también guardado en el repositorio de GIT o del sistema empleado.	5
Funcionamiento del programa. Usted adjunta también screenshots o videos para mostrar que el programa funciona.	15
Funcionamiento del programa usando la clase implementada por otros grupos. El auxiliar seleccionará estos grupos al azar, pero usted puede indicar con quienes ha probado que funciona su programa. Adjunte screenshots de funcionamiento de su programa funcionando con clases implementadas por otros grupos.	15
Su clase de radio funciona en el programa de otros grupos. El auxiliar utilizará la clase radio que su grupo elaboró y lo probará con el programa principal de otro grupo seleccionado al azar. Adjunte screenshots de funcionamiento del programa de otros grupos que emplean las clases desarrolladas por usted.	15
Utiliza JUnit para probar por lo menos tres de las operaciones del radio. Adjunte con su tarea las pruebas unitarias definidas. Adjunte screenshots de la ejecución de las pruebas. Debe existir por lo menos una muestra de que las pruebas fallan y una de que las pruebas si funcionan.	20
TOTAL:	100

[1] Ya están antiguas, pero aún tienen puntos útiles: http://www.oracle.com/technetwork/java/codeconvtoc-136057.html
[2] http://google-styleguide.googlecode.com/svn/trunk/javaguide.html