PROYECTO
EJERCICIO DE CÁTEDRA N° 3

FECHA DE ENTREGA AL ESTUDIANTE: 9 de junio
FECHA DE ENTREGA AL DOCENTE: 24 de junio, hasta antes de la medianoche

CONDICIONES DEL PROYECTO:

El ejercicio #3 del Portafolio de Aprendizaje del curso EIF200 Fundamentos de Informática tiene como propósito que el estudiante ponga en práctica la resolución de problemas de programación utilizando clases y arreglos bidimensionales en C++, según lo establecido por la cátedra.

El ejercicio se debe trabajar en parejas, sólo uno de los estudiantes deberá subir la solución en la actividad correspondiente en el aula virtual. El nombre del archivo comprimido debe incluir el número de ejercicio del portafolio, el número de grupo y el nombre de ambos estudiantes. Por ejemplo: Port2G10RosaRyJuanA.rarm. Incluya en la carpeta un bloc de notas con el nombre completo y el número de cédula de cada participante.

Cualquier copia o plagio entre grupos o estudiantes se calificará con 0, como lo establece el artículo 24 del Reglamento General sobre los Procesos de Enseñanza y Aprendizaje de la Universidad Nacional.

Para este ejercicio se debe adjuntar la documentación tal como se especifica en el documento **Normas para la presentación de proyectos**, disponible en el Aula Virtual. Queda a criterio de cada profesor solicitar la defensa del proyecto.

DESCRIPCIÓN DEL PROBLEMA:

Los vehículos eléctricos son aquellos vehículos propulsados por energía eléctrica almacenada en sus baterías internas. El primer coche eléctrico se remonta al año 1828 gracias al proyecto diseñado por Ányos Jedlik. Fue en 1899 cuando un vehículo eléctrico, conocido como "La Jamais Contente", fue capaz de alcanzar por primera vez la velocidad de 100 km/h[1].

El carro eléctrico ayuda a conservar el medio ambiente al no emitir gases de efecto invernadero. El vehículo convencional de combustión interna, por su parte, produce dióxido de carbono, una de las principales causas del calentamiento global, al quemar el combustible fósil, sea gasolina o diésel. Los principales contaminantes producidos por los vehículos de combustión incluyen: dióxido de carbono, monóxido de carbono, óxidos nitrosos, compuestos orgánicos volátiles, material particulado y azufre.

Los carros eléctricos funcionan haciendo uso de baterías eléctricas, estas son de diferentes tipos: Plomo-ácido (PB-ácido), Níquel-cadmio (NiCd), Níquel-hidruro metálico (NiMh), Ion-litio (LiCoO2), Ion-litio con cátodo de LiFePO4, Polímero de litio (LiPo). Cada batería eléctrica está compuesta de celdas o celdillas, las cuales son revisadas en los talleres especializados para conocer su condición y el momento en el cual deben cambiarse.

[1] https://www.lugenergy.com/que-es-vehiculo-electrico/

Costa Rica es un país con un alto compromiso ambiental que ocupa el tercer lugar en Latinoamérica como país con más puntos de carga para autos eléctricos. Suponga que los estudiantes de la Universidad Nacional de Costa Rica han inventado una nueva batería para ser usada en motores eléctricos, dicha innovación promete revolucionar el mercado de carros a nivel nacional e internacional, promoviendo así una verdadera revolución ambiental.

Los estudiantes de la Escuela de Informática han participado activamente en el desarrollo de esta nueva tecnología y ahora tienen a cargo la tarea de realizar un programa que permita registrar y analizar las celdas de las baterías en desarrollo.

Al crear una nueva batería se debe registrar un código de fabricación de la batería, el tipo (mencionado anteriormente), el modelo, el año de fabricación, el costo de producción y una colección de celdas o celdillas eléctricas, cada celda o celdilla tiene registrados:

- **Condición:** número entero cuyo valor es **1** para una celda en buen estado o **0** para una celda dañada y, **-1** para una celda no habilitada en la matriz.
- **Voltios:** número entero entre 0 y 12 que representa la tensión o diferencia de potencial de la celda.
- **Amperios:** número entero entre 0 y 12 que representa la intensidad de corriente de la celda.

Con esta nueva tecnología es posible fabricar baterías con una colección de celdas de diferentes tamaños, de hasta 25x25 celdas. Es decir, toda nueva batería tiene una capacidad máxima de 25x25 celdas, pero no necesariamente en su creación deben habilitarse todas las celdas posibles. Pueden crearse baterías con matrices de celdas de tamaños inferiores, por ejemplo 12x12, 14x3, 20x11, 15x20, etc, pero nunca excederse de 25x25.

Ejemplo:

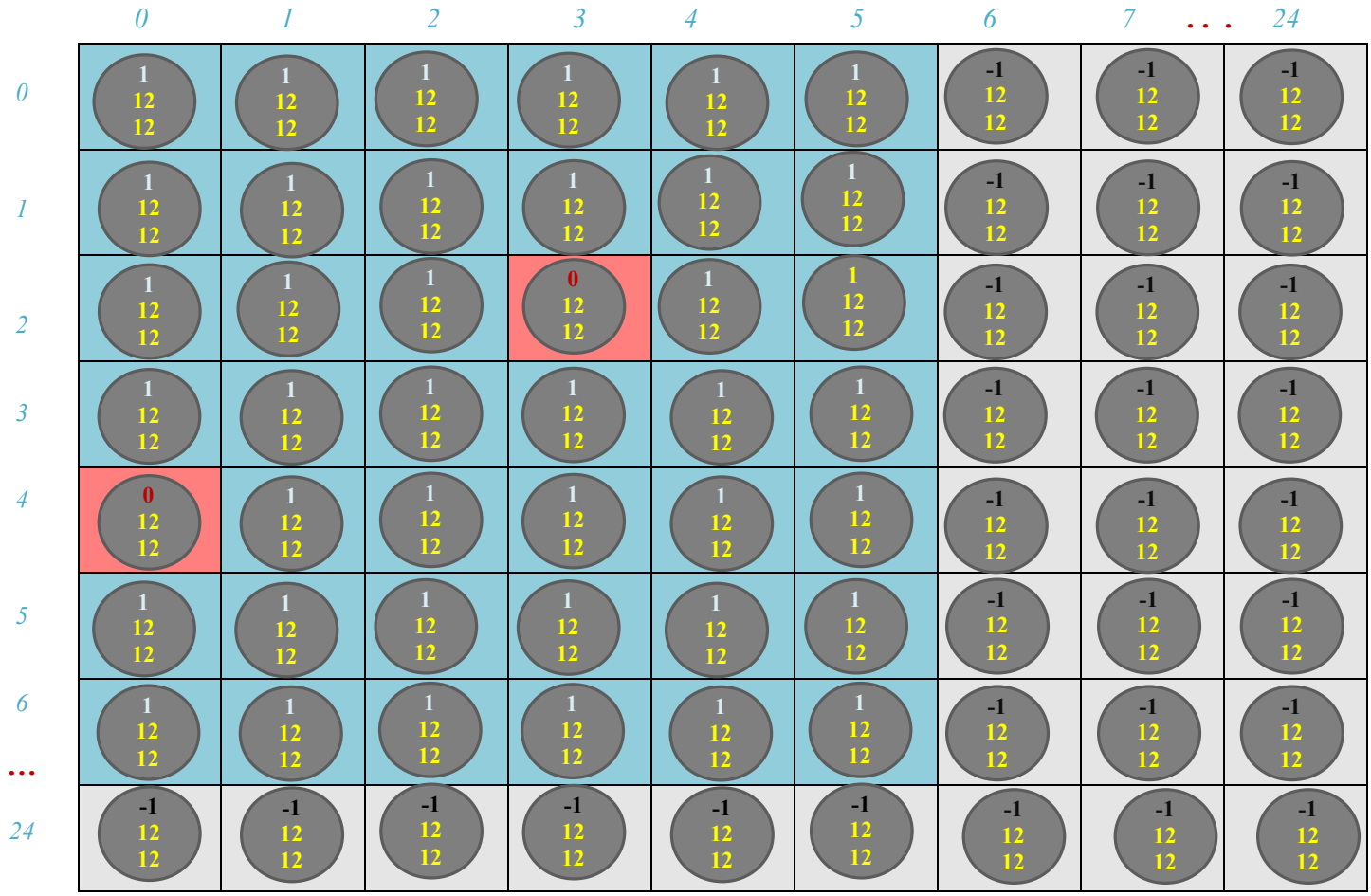
El siguiente dibujo representa el ejemplo de una batería con una matriz de celdas o celdillas eléctricas de 25x25.

Note que no todas las celdas de la batería se encuentran habilitadas, a pesar de que la matriz de celdas es de 25x25. En este ejemplo, solamente están habilitadas las celdas de las 7 primeras filas y las 6 primeras columnas (7x6).

Observe además que dicha matriz tiene dos celdas dañadas: celda (4, 0) y celda (2, 3).

En el dibujo, cada círculo de la matriz representa una celda. Los números que se muestran en cada círculo representan los valores de los atributos de dicha celda (condición, voltios y amperios).

- "**Voltios:** número entero entre 0 y 12 que representa la tensión o diferencia de potencial de la celda. **Amperios:** número entero entre 0 y 12 que representa la intensidad de corriente de la celda."



Al crear una batería se debe recibir del usuario del programa la cantidad de filas y columnas de celdas que se habilitarán en la matriz, pues, como se indicó anteriormente, no en todas las baterías se habilita la totalidad de las celdas posibles. Se debe considerar que una nueva batería inicia con todas sus celdas activas, es decir, en ***condición*** 1. Las celdas no habilitadas o no activas, tendrán ***condición*** igual a -1.

Como principio de fabricación, cada nueva batería tendrá el mismo valor en voltios y amperios para TODAS sus celdas habilitadas. No obstante, estos valores pueden cambiarse posteriormente, de modo tal que cada batería pueda llegar a tener celdas con valores diferentes de ***voltios*** y ***amperios***.

Al iniciar el programa se deberá crear dos baterías de prueba ***b***1 y ***b2.*** El valor de los atributos de estos objetos será establecido por el usuario.

Una vez creadas las dos baterías de prueba se deberá presentar al usuario el siguiente menú de opciones para realizar análisis de las baterías.

Menu

(1) Información de la batería
(2) Detalle de una celda
(3) Llenar aleatoriamente las celdas de una batería
(4) Modificar una celda
(5) Determinar el estado de la batería
(6) Modificar la capacidad de la batería
(7) Intercambiar las baterías
(8) Comparar baterías
(0) Salir

Se deben implementar los siguientes métodos para una Batería:

1. (10pts) **Método toStringBateria** *(opción #1 del menú)*. Este método deberá retornar la información general de cada batería. Para cada celda de la matriz solo se deberá incluir su condición (0,1,-1). Un ejemplo del despliegue de los datos que devuelve este método es el siguiente:

```
 Datos de la Bateria:
    Tipo:7
    Modelo:7
    Annio:33
    costo de produccion:33

    Filas habilitadas:7
    Columnas habilitadas:7

 1  1  1  1  1  1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
 1  1  1  1  1  1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
 1  1  1  1  1  1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
 1  1  1  1  1  1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
 1  1  1  1  1  1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
 1  1  1  1  1  1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
 1  1  1  1  1  1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
```

Nota: En el despliegue en pantalla se debe mostrar la matriz completa de 25x25 posiciones (cada línea representa una fila de la matriz). El número que se muestra en cada posición corresponde con la "***condición***" de la celda correspondiente (0, 1, -1).

2. (10 pts) **Método toStringDetalleCelda** *(opción #2 del menú)*. Este método recibe una posición de la matriz, por fila y columna, y devuelve el detalle completo de la celda correspondiente, tal como se muestra en el siguiente ejemplo para la posición [1, 1]:

Datos de la celda en la posición [1, 1]:

<Valores de los atributos de la celda correspondiente>

Este método debe utilizar el método toString() de la clase Celda.

3. (10 pts) **Método llenarAleatorio** *(opción #3 del menú)*. Este método asigna valores generados aleatoriamente a los atributos de una batería y de cada una de sus celdas habilitadas. Al invocar este método se dispondrá de datos de la batería y de sus celdas para poder realizar análisis.

4. (10 pts) **Método modificarCelda** *(opción #4 del menú)*. Este método recibe una posición habilitada de la matriz, por fila y columna, y un objeto de la clase Celda para registrarlo en esa posición de la matriz. Con este método se pueden modificar los valores de los atributos de la celda correspondiente.

5. (10 pts) **Método estadoDeLaBatería** *(opción #5 del menú)*. Este método debe determinar y preparar una hilera para mostrar la cantidad de celdas en buen estado *(condición 1)*, la cantidad de celdas dañadas *(condición 0)*, la cantidad de celdas inhabilitadas *(condición -1)*, el porcentaje de celdas habilitadas en buen estado, el porcentaje de celdas habilitadas dañadas y, el porcentaje de celdas inhabilitadas. Es importante aclarar que el porcentaje de las celdas inhabilitadas se debe calcular con base en el total de celdas posibles de la batería que son: 625 = 25 * 25.

6. (10 pts) **Método modificarTamanoBateria** *(opción #6 del menú)***.** Este método permite modificar la cantidad de filas y columnas de celdas habilitadas en la matriz de celdas de la batería. El método debe recibir la nueva cantidad de filas y de columnas de la matriz, las cuales deben estar en el rango de 1 a 25. Si se agregan nuevas celdas, el valor de sus atributos ***voltios y amperios*** se deben recibir del usuario y el valor de la ***condición*** de dichas celdas deberá ser 1. No se debe alterar el valor de los atributos de las otras celdas de la matriz.

   Al invocar este método se debe verificar que la cantidad de filas y de columnas estén dentro del rango correspondiente.

7. (10 pts) **Método cambiarBateria** *(opción #7 del menú)***.** Este método recibirá como parámetro una batería e intercambiará los datos de la batería del objeto propio y de sus celdas por los datos correspondientes de la batería recibida como parámetro. Los cambios realizados por el método en ambos objetos deben prevalecer una vez finalice el método.

8. (10 pts) **Método compararBaterias** *(opción #8 del menú)*. Este método debe recibir como parámetro una batería y devolver el promedio de ***amperios*** de las celdas de cada batería: de la propia y de la recibida como parámetro. Con base en los valores retornados por el método se debe mostrar en la pantalla el promedio de amperios de cada batería y cuál de las 2 tiene el mayor promedio. Con base en los valores retornados por el método se debe mostrar en la pantalla el promedio de amperios de cada batería y cuál de las 2 tiene el mayor promedio

9. (20 pts) El programa debe realizar las validaciones necesarias para las diferentes entradas del usuario, de tal forma que se acepten solamente valores válidos y coherentes para cada atributo, parámetro u opción del menú. Además debe hacer uso eficiente de las estructuras de programación y del main() (defina funciones para el uso del menú, la creación de instancias de las clases, etc), las funciones y métodos deben realizar un solo objetivo. No se le pide documentación externa, pero la interna si, a razón de que se pueda entender lo realizado.

En la evaluación de cada uno de los métodos se considerará el cumplimiento o consistencia con cada uno de los aspectos descritos en este enunciado.

CONSIDERACIONES:

1. Cree la(s) clase(s) que permita(n) representar una batería eléctrica, tal como se describe en este enunciado. Incluya correctamente los atributos requeridos e implemente correctamente todos los métodos básicos de la(s) clase(s). Utilice el encapsulamiento adecuado en la(s) clase(s) necesarias(s) para solucionar el ejercicio.

2. La interface con el usuario debe ser clara, informativa y amable. Se debe utilizar una interfaz agradable y bien diseñada, sin errores ortográficos.

3. Utilice eficientemente las estructuras de programación y evite la repetición innecesaria de código.

4. Todas las llamadas a los métodos deben hacerse con parámetros válidos. Por lo tanto, se deben realizar las validaciones previas que se requieran.

5. Ningún método de una clase debe interactuar con el usuario (imprimir/desplegar o leer). Estas acciones se deben realizar en el main() o en una función externa a las clases.

6. El programa no debe tener errores de sintaxis y debe tener la funcionalidad descrita en este enunciado.