Parte 2

Clase	Responsabilidad
StartupBust	Se encarga de ejecutar todo el programa pues tiene el método main, además, se encarga de establecer la configuración inicial del juego con el método setUpGame(), también, es el encargado de permitir que el usuario juegue, pues con el método startPlaying() le permite ingresar una suposición y con el método checkUserGuess(), verifica si la suposición del usuario es correcta o no. Finalmente, también se encarga de mostrar un formato en pantalla cuando el juego finalice, con el método finishGame().
Startup	Se encarga de generar los objetos tipo Startup que serán los barcos u objetivos a disparar, esto lo realiza asignándole a cada objeto unas celdas donde estará localizado y un nombre, con los métodos setLocationCells() y setName() respectivamente. Por otro lado, con el método checkYourself(), puede determinar si el objeto fue atinado por la suposición del usuario, si falló o si ya le atinó a todas sus posiciones, este método lo llama la clase StartupBust para con cada suposición del usuario, hacer la verificación de si le atinó a cada objetivo.
GameHelper	Esta clase se encarga de obtener la suposición del usuario con el método getUserInput(), además, con el método palceStartup() devuelve el listado de celdas para posicionar cada objeto de tipo Startup, en éste método hace uso de el resto de métodos para poder lograr su objetivo, por ejemplo: - startupFits(): Mira si la posición final está bien ubicada, para que cada Startup tenga su ubicación en tres celdas consecutivas, ya sean verticales u horizontales. - coordsAvailable(): Verifica si cada una de las coordenadas asignadas al objeto está disponible, de lo contrario retorna false para que vuelva a crearlas.

GameHelper	- savePositionToGrid(): Se encarga de que una vez las coordenadas estén bien, ocupen nuestra matriz de coordenadas en su respectivo índice. - ConvertCoordsToAlphaFormat(): Toma las coordenadas creadas y las transforma en coordenadas alfanuméricas, por ejemplo A1, E3, C2, etc, esto haciendo uso del método getAlphaCoordsFromIndex(). - getAlphaCoordsFromIndex(): Haciendo uso del índice que fue asignado a la coordenada, se determina su fila con el método calcRowFromIndex() y también obtiene la columna a partir de su índice, luego con la columna y el atributo de la clase ALPHABET, logra obtener el nombre alfabético de la columna, finalmente retorna la letra unida al número. - calcRowFromIndex(): Haciendo uso del índice, determina la fila a la que pertenece. - getIncrement(): Se encarga de ver si el objeto es ubicado horizontal o verticalmente.

Parte 3

Deficiencias:

- Interfaz de usuario en español.
- Después de cada disparo debe informar el status del juego (Total de disparos y lista de impactos)
- No procesar disparos por fuera del tablero
- Numerar las columnas del tablero desde 1.
- La clase GameHelper tiene muchas responsabilidades (podría dividir en dos esta clase)
- El método Main no se encuentra en una clase independiente.

Parte 4

Descripción del Problema identificado	Descripción de cambio realizado
En el archivo original, la interfaz con la que interactúa el usuario se veía en inglés y no en español.	Cada print que tuviera el programa y cada salida que se mostrara al usuario se cambió del inglés al español.
Cuando el usuario ingresa una suposición, solo se muestra si acertó o falló, más no se muestra cuántos disparos lleva acertados ni cuántos barcos lleva hundidos.	Se agregó en la clase StartupBust una variable tipo int llamada numOfGoods, que cuenta cuántos aciertos tiene y en el método checkUserGuess, en el condicional que verifica si la suposición del usuario fue correcta se agregó que esta variable aumentara en 1 unidad, también se agregó que aumentara en 1 unidad si destruía el barco. Además, en éste mismo método al final del ciclo for, se imprime el informe con los disparos que ha realizado (numOfGuesses) y con los aciertos que ha tenido (numOfGoods).
En el programa inicial, el usuario podía ingresar suposiciones fuera de las posibilidades, es decir, aunque el programa solo identificaba letras desde la “a” a la “g” y números del 0 al 6, el usuario podía ingresar valores como z9 o a8, y el programa no le decía nada más que “miss”.	En la clase Startup, se agregó un objeto tipo GameHelper para extraer el alfabeto y la cantidad de columnas que se usarán y almacenarlos en las variables tipo String alphabet y tipo int intLimit. En el método checkYourself, se agregaron las variables tipo String letter y number, que extraen la letra y el número de la userInput respectivamente. Luego, en el mismo método, con un condicional verificamos que la letra extraída esté en el alfabeto que usaremos y que el número extraído esté en el intervalo cerrado

	[1,intLimit], de ser así, que haga la verificación, de lo contrario que pida un dato válido.
El programa, aunque tenía la celda de juego de tamaño 7x7, las entradas se validaban desde 0 hasta 6.	Se hizo la corrección en la clase GameHelper, pues aquí se construyen las posiciones que tomará cada uno de los barcos; para el cambio, se editó el método getAlphaCoordsFromIndex(), ya que éste es usado para construir la coordenada de cada barco, como a0, a1, a2, etc, entonces lo que estaba fallando era cómo se agregaba el valor numérico que corresponde a la fila, el cual se estaba calculando con el método calcRowFromIndex() que divide a cada índice entre la longitud de la celda, entonces si el índice era 0, lo dejaba como 0, es por eso que cuando se calcula la variable row en el método getAlphaCoordsFromIndex(), se agregó un +1 para evitar que inicie en 0 y asegurar que termine en 7.
Antes de editar el programa, la clase GameHelper tenía asociadas muchas responsabilidades.	Como GameHelper tenía muchas responsabilidades, se creó la clase HelperOfHelper que hace todos los procesos internos y pequeños que antes hacía GameHelper, actualmente GameHelper solo se encarga de obtener la entrada del usuario con el método getUserInput() y de ubicar cada barco con el método placeStartup(). Por otro lado, la clase HelperOfHelper, se quedó con los métodos usados por GameHelper en el método palceStartup(), mostrados a continuación: - startupFits() - coordsAvailable() - savePositionToGrid() - convertCoordsToAlphaFormat() - getAlphaCoordsFromIndex() - calcRowFromIndex() - getIncrement() - getAlphabet() - getGrid() - getStartupCount(): Éste se creó con el fin de usar el mismo startupCount en la clase HelperOfHelper y en la clase GameHelper. - getGridSize(): Al igual que el anterior, se creó con el mismo fin pero para GRID_SIZE.

	También se quedó con los atributos: - ALPHABET - GRID_LENGTH - GRID_SIZE - HORIZONTAL_INCREMENT - VERTICAL_INCREMENT - Grid - startupCount Y a algunos que se ocupaban en otras clases, se les creó sus métodos getter, como es el caso de startupCount y GRID_SIZE. De éste modo, la clase GameHelper se encarga de procesos más grandes y la clase HelperOfHelper se encarga de los procesos pequeños que antes le correspondían a GameHelper.
El método Main no se encuentra en una clase independiente.	Aunque no representa un gran error, se considera parte de las buenas prácticas de programación dejar el método main en una clase aparte y un paquete distinto, es por eso que se creó el paquete ui y la clase Main en ese paquete, donde se copió el método main que estaba en la clase StartupBust, además, para esto fue necesario cambiar el encapsulamiento de los métodos en la clase StartupBust, setUpGame() y startPlaying() de privado a público.