**Informe práctica 2: Hundir la flota**



Fernando Pérez 1493076

Pol Miranda 1423191

2019-2020

Test i Qualitat del Software

**Tabla de contenidos**

1. Introducción
2. Exploratory testing
   1. Ataque 1
      1. Descripción
      2. Resultado
   2. Ataque 2
      1. Descripción
      2. Resultado
   3. Ataque 3
      1. Descripción
      2. Resultado
   4. Ataque 4
      1. Descripción
      2. Resultado
   5. Ataque 5
      1. Descripción
      2. Resultado
   6. Ataque 6
      1. Descripción
      2. Resultado
   7. Ataque 7
      1. Descripción
      2. Resultado
   8. Ataque 8
      1. Descripción
      2. Resultado
   9. Ataque 9
      1. Descripción
      2. Resultado
   10. Ataque 10
       1. Descripción
       2. Resultado
   11. Ataque 11
       1. Descripción
       2. Resultado
   12. Ataque 12
       1. Descripción
       2. Resultado
   13. Ataque 13
       1. Descripción
       2. Resultado
   14. Ataque 14
       1. Descripción
       2. Resultado
3. RTF
   1. Inspection Summary Report
   2. Inspection Issue Log
   3. Typo List
4. Test Cases
   1. Escenario 1
   2. Escenario 2
   3. Escenario 3
   4. Escenario 4
   5. Escenario 5
   6. Escenario 6
   7. Escenario 7
   8. Escenario 8
5. **Introducción**

En la segunda parte de la práctica debemos realizar el testing de uno de los juegos desarrollados por nuestros compañeros en la primera parte del curso, para ello aplicaremos diversas técnicas vistas en clase.

En nuestro caso hemos seleccionado el proyecto basado en el juego *Hundir la flota*, a continuación se verá las diversas técnicas que hemos aplicado para realizar el testeo de la aplicación.

La primera técnica aplicada ha sido el Exploratory testing, donde a partir de los ataques vistos en el curso se ha obtenido un cierto feedback del programa, hay que aclarar que no se han podido realizar todos los ataques ya que el juego no lo permite.

Seguidamente mostraremos el RTF que realizamos juntamente con otros alumnos que nos ayudaron en la tarea. En esta parte detectamos ciertos errores en el código ya sea de diseño, construcción, *checkstyle*, requerimientos, etc.

En la última parte del documento se muestran los *test cases*, donde se han propuesto una serie de escenarios para probar el funcionamiento del programa y ver cómo responde a los diferentes casos propuestos.

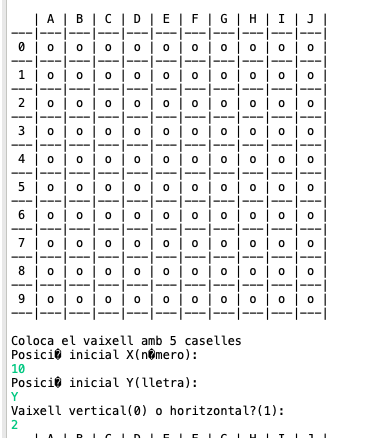
Para concluir la práctica, también se han realizado test automatizados a una serie de métodos a bajo nivel, esta parte no está incluida en el informe.

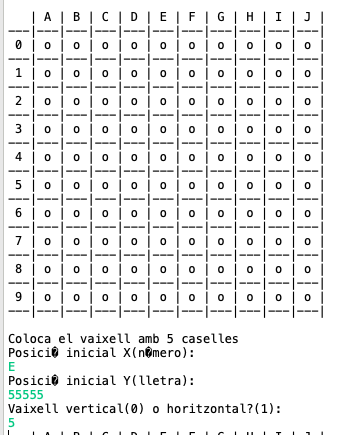
**2. Exploratory testing**

**2.1 Ataque 1**

**2.1.1 Descripción**

Aplicar entradas que obliguen a que todos los mensajes de error ocurran al menos una vez

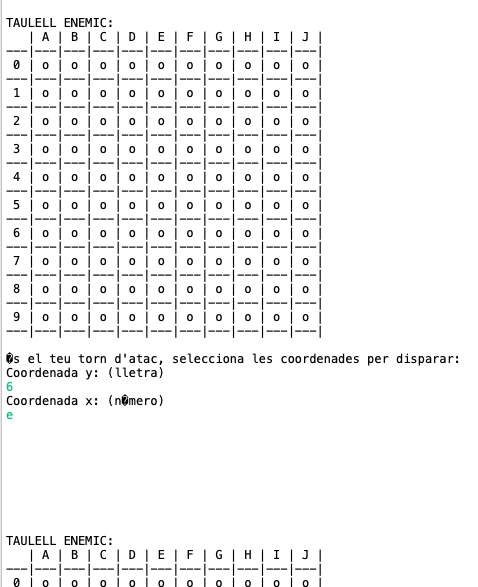
**2.1.2 Resultado**



1. Captura de pantalla de inputs incorrectos(1)

1.Captura de pantalla de inputs incorrectos(2)

Cuando introducimos una posición inicial *X* e *Y* (una posición del barco incorrecta) no nos sale ningún tipo de mensaje de error y se puede seguir jugando al juego. Por lo tanto podemos decir que el código puede soportar la entrada de inputs no válidos.



2. Captura de pantalla de inputs incorrectos y la respuesta ante estos

De la misma manera, cuando jugamos contra la IA, aunque pongamos un carácter que no sea correcto podemos seguir jugando sin que nos muestre ningún mensaje de error. Aunque no nos salga un mensaje de error perdemos un turno el cual la IA aprovecha para atacar (esto se ve reflejado en el TC2\_002).

**2.2 Ataque 2**

**2.2.1 Descripción**

 Aplicar entradas que obliguen a establecer valores por defecto.

**2.2.2 Resultados**

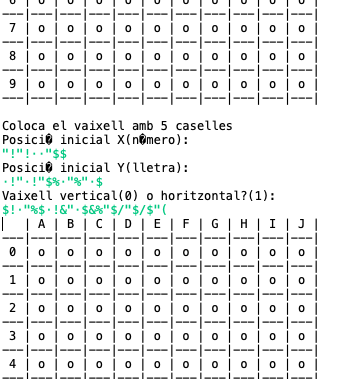
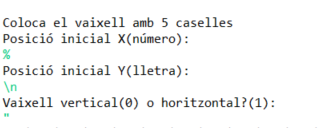
Dado que los únicos inputs que podemos poner son las coordenadas y no, por ejemplo, la creación de un tablero (que en ese caso tendría unos valores máximos y mínimos por defecto) no se puede realizar este ataque.

**2.3 Ataque 3**

**2.3.1 Descripción**

Explorar conjunto de caracteres permitidos y valores de significado potencialmente especiales en campos de cadena

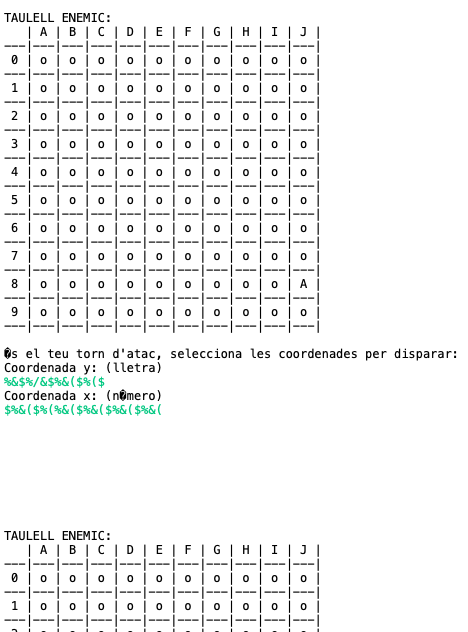
**2.3.2 Resultados**



3.Captura de pantalla de introducción de carácteres especiales(1)

4.Captura de pantalla de introducción de carácteres especiales(2)

Como se puede ver al introducir caracteres o cadena de caracteres de significado especiales no devuelve ningún mensaje de error y se puede seguir jugando al juego. Los inputs están muy bien controlados para este tipo de situaciones, el juego seguirá pidiendo coordenadas y orientación del barco hasta obtener caracteres correctos.



5.Captura de pantalla de introducción de carácteres especiales al atacar al tablero enemigo

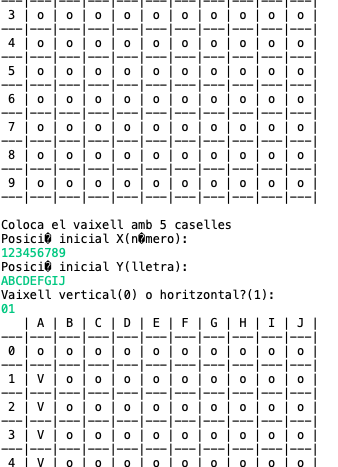
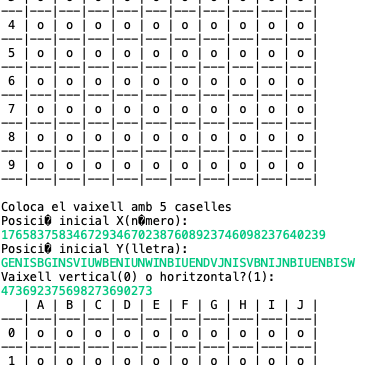
También nos encontramos con el mismo caso a la hora de jugar contra la IA, pero perdiendo nuestro turno.

**2.4 Ataque 4**

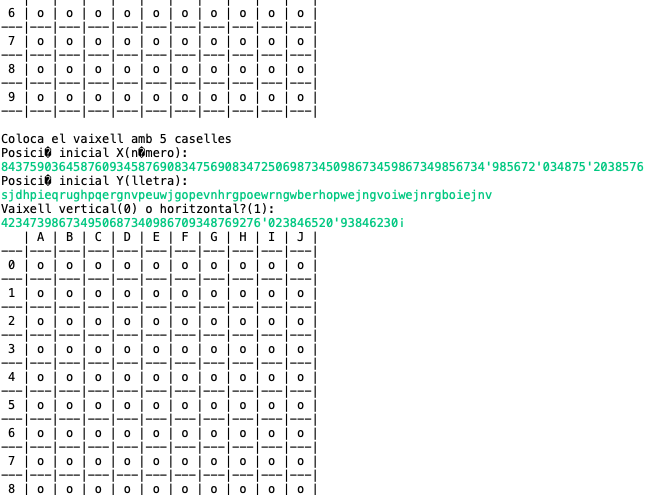
**2.4.1 Descripción**

Desbordamiento de buffers de entrada o cadena de parámetros

**2.4.2 Resultados**

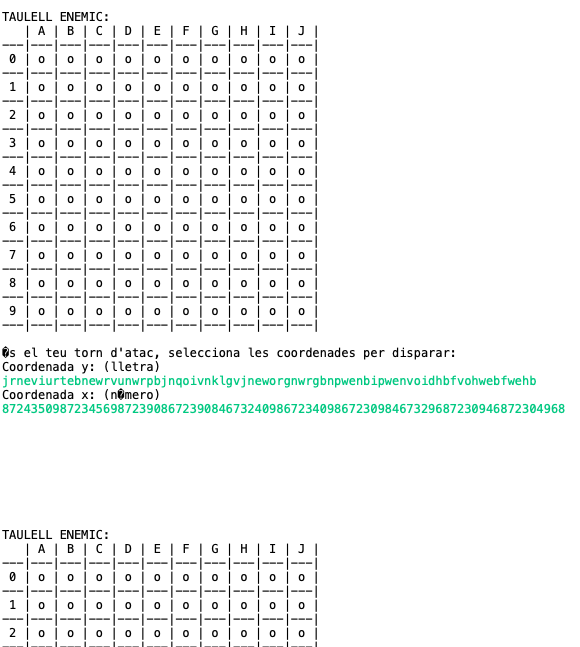
 7.Captura de pantalla de desbordamiento de valores en los inputs(2)

6Captura de pantalla de desbordamiento de valores en los inputs(1)



8Captura de pantalla de desbordamiento de valores en los inputs(3)

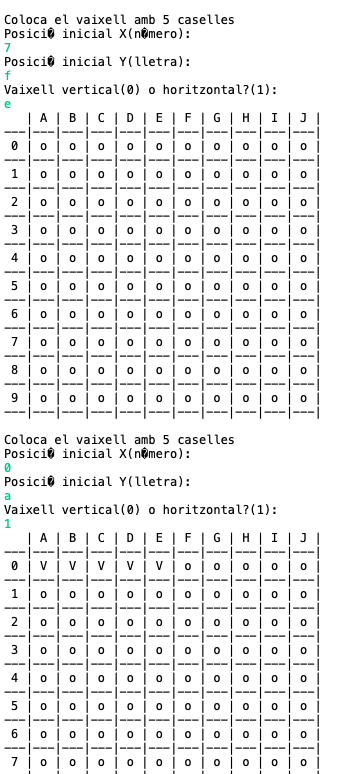
Por muy larga que sea la cadena de caracteres numéricos o alfabéticos que se introduzca se sigue sin mostrar ningún tipo de mensaje de error ni ninguna excepción. El juego se sigue ejecutando y pidiendo nuevos inputs válidos.



9Captura de pantalla de desbordamiento de valores en los inputs en el turno de atacar

También nos encontramos con el mismo caso a la hora de jugar contra la IA.

**2.5 Ataque 5**



**2.5.1 Descripción**

10.Captura de pantalla combinando inputs(1)

Encontrar entradas que puedan interactuar y probar combinaciones de sus valores.

**2.5.2 Resultados**

Si introducimos valores no válidos, una vez se introduce un número y una letra válidas, éstas se colocan correctamente en el tablero.

11.Captura de pantalla combinando inputs(2)

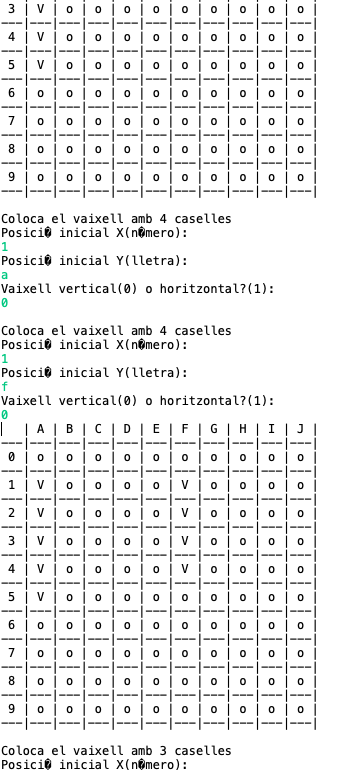
|  |  |
| --- | --- |
| https://lh5.googleusercontent.com/yJETAsksFiNfSp0geyGTBPRaQRCFN6w3AajQYKu_TBya7Wko1SR6Ur5GArCykimaEnZaspiJikdFz0IchRPsfLWSiF7s1XWN5sPdRfli1HLUh4JLVSR8Nm6yHmsDPhcKS-kJH2yS  12.Captura de pantalla combinando inputs(2) | Si introducimos valores no válidos, una vez se introduce un número y una letra válidas, éstas se colocan correctamente en el tablero. |

**2.6 Ataque 6**

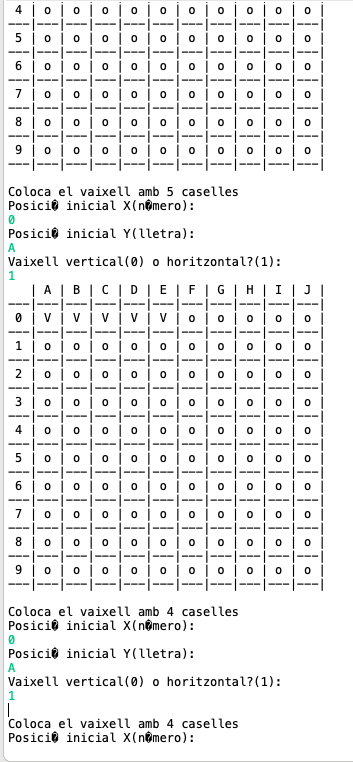
**2.6.1 Descripción**

Repetir la misma entrada o serie de entradas varias veces

**2.6.2 Resultados**

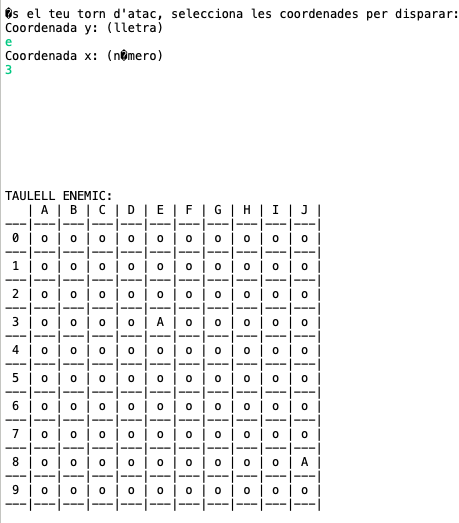
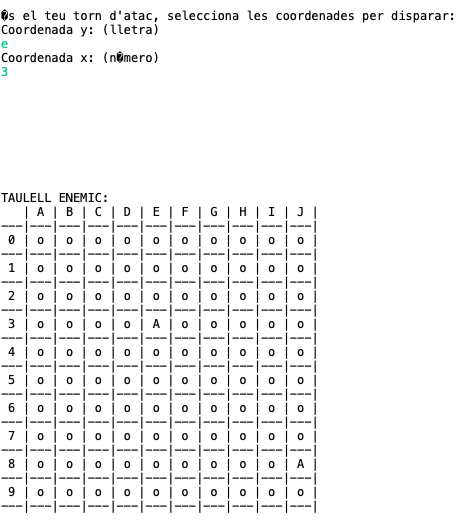


13.Captura de pantalla de la repetición de la colocación de un barco(2)



14.Captura de pantalla de la repetición de la colocación de un barco(1)

Cuando introducimos la misma coordenada para posicionar un barco, el tablero desaparece. Hasta que no volvemos a introducir una coordenada correcta no se nos vuelve a mostrar.



15. Captura de pantalla de la repetición de un ataque(2)

16. Captura de pantalla de la repetición de un ataque(1)

En el caso de jugar contra la IA si ponemos en dos turnos seguidos la misma coordenada el tablero se nos sigue mostrando y podemos seguir con la partida.

**2.7 Ataque 7**

**2.7.1 Descripción**

Conocer el dominio del problema y pensar en casos especiales de combinaciones de entrada que obligan a generar salidas no válidas.

**2.7.2 Resultados:**

No se ha podido realizar este ataque ya que las salidas están limitadas y programadas por defecto.

**2.8 Ataque 8**

**2.8.1 Descripción**

Forzar propiedades de una salida para cambiar.

**2.8.2 Resultados**

Dado que todas las constantes (tamaño de los barcos, tamaño del tablero, tipo de outputs, etc.) no se pueden modificar durante la ejecución del juego, este test no se ha podido realizar de manera satisfactoria.

**2.9 Ataque 9**

**2.9.1 Descripción**

Forzar la actualización de la pantalla para encontrar problemas de renderizado en aplicaciones con salida gráfica.

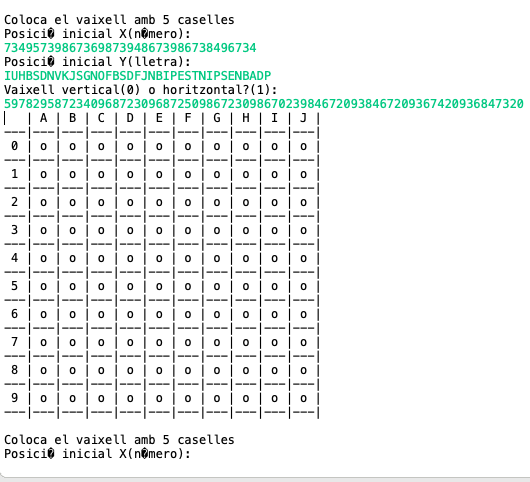
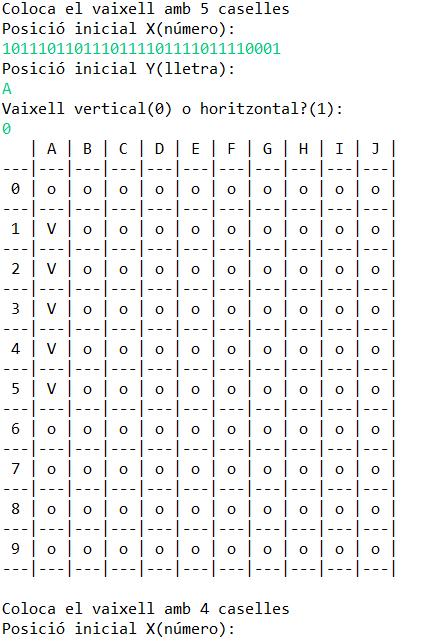
**2.9.2 Resultados**

Los outputs están controlados por la consola así que no hemos podido encontrar ningún error.

**2.10 Ataque 10**

**2.10.1 Descripción**

Forzar estructuras para almacenar demasiados o muy pocos valores.

**2.10.2 Resultados**

17.Captura de pantalla de un input grande(2)

18.Captura de pantalla de un input grande(1)

Aunque introducimos valores y cadenas de caracteres muy grandes el programa sigue funcionando pero sin mostrar ningún mensaje de error. En la imagen 18 pero, podemos ver que si introducimos caracteres válidos, pero siendo demasiados grandes, selecciona solo el primer carácter.

**2.11 Ataque 11**

**2.11.1 Descripción**

Investigar formas alternativas de modificar las restricciones internas en las propiedades de los datos además del tamaño en la creación de la estructura de datos.

**2.11.2 Resultados**

El código no nos deja manipular el tamaño del tablero/barcos ya que automáticamente lo hace, por lo tanto no hemos podido realizar este ataque.

**2.12 Ataque 12**

**2.12.1 Descripción**

Experimentar con combinaciones de operandos y operadores que pueden hacer que el software falle.

**2.12.2 Resultados**

No se ha podido realizar este ataque ya que todos los caracteres no permitidos se omiten y se vuelve a pedir un input hasta que sea correcto.

**2.13 Ataque 13**

**2.13.1 Descripción**

Forzar una función para llamarse a sí misma recursivamente.

**2.13.2 Resultados**

No se ha podido realizado este ataque.

**2.14 Ataque 14**

**2.14.1 Descripción**

Encontrar características que comparten datos o interactúan pobremente.

**2.14.2 Resultados**

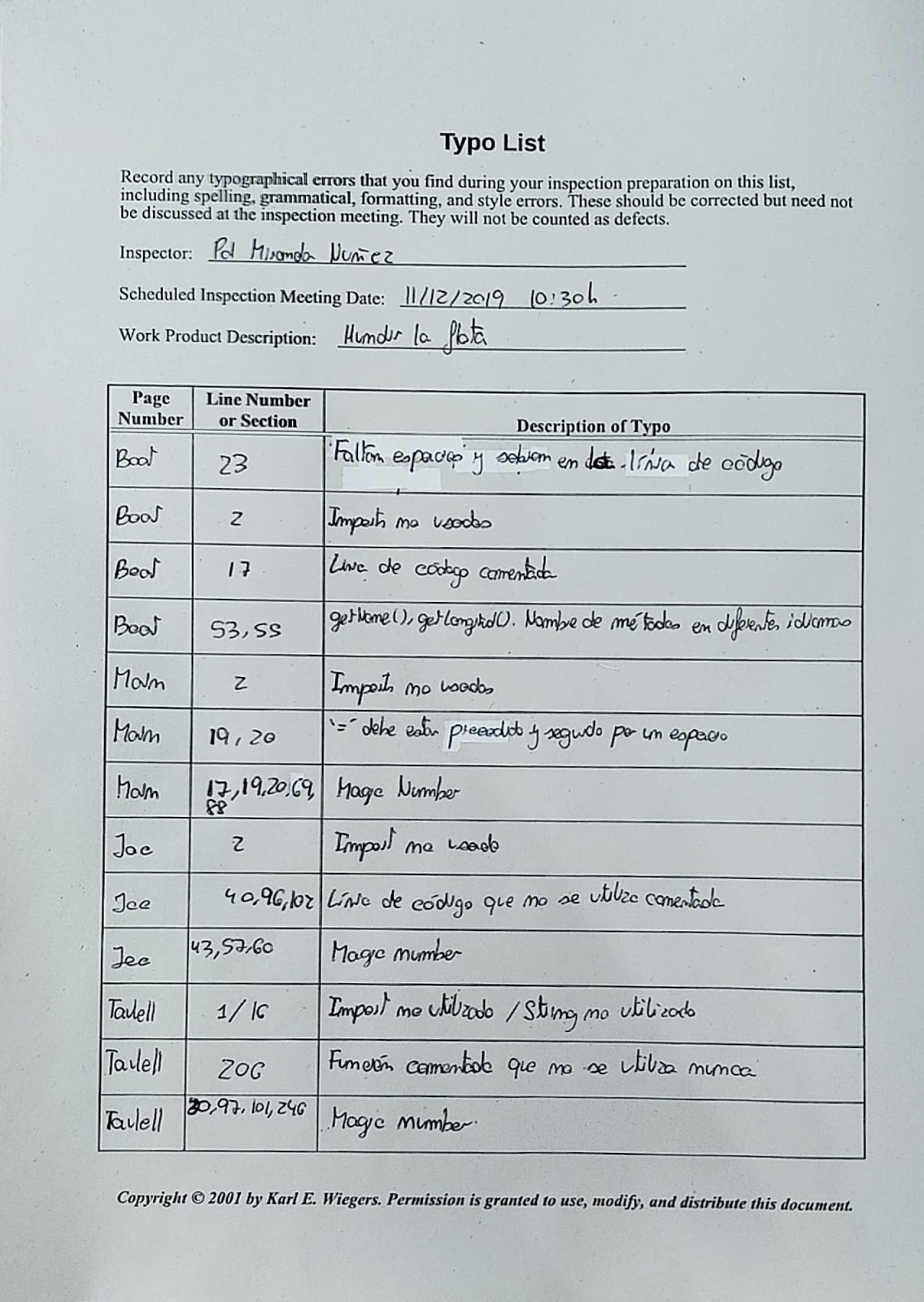
No se ha podido realizado este ataque.

**3. RTF**

3.1 Inspection Summary Report

3.2 Inspection Issue Log

3.3 Typo List



4. Test Cases

4.1 Escenario 1