UBA – Facultad de Ciencias Exactas y Naturales – Departamento de Computación

Algoritmos y Estructuras de Datos I

Segundo cuatrimestre de 2022

Versión 2.0 (11 de septiembre de 2022)

Trabajo Práctico de Especificación

Buscaminas

1. Modalidad de Trabajo

- El Trabajo Práctico se realiza en grupo de 4 personas. Todos deben pertenecer al mismo turno.
- Para aprobar el trabajo se necesita:
  - Que todos los ejercicios estén resueltos.
  - Que las soluciones sean correctas.
  - Que la solución sea declarativa.
  - Que el lenguaje de especificación esté bien utilizado.
  - Que las soluciones sean prolijas: evitar repetir especificaciones innecesariamente y usar adecuadamente las funciones y predicados auxiliares.
- Entregable:
  1. Completar la solución de los ejercicios usando LaTeX y siguiendo el *template* dado por la cátedra.
  2. Entregar la versión digital de la resolución en formato **.pdf** a través del campus de la materia por cualquiera de los miembros del grupo (realizar **una** sola entrega por grupo).

2. Introducción

Durante mucho tiempo, si uno estaba aburrido y tenía una compu a mano, seguramente terminaba jugando al buscaminas. Era un juego simple y adictivo que se encontraba en todas las computadoras personales, sin importar qué sistema operativo se usara.

3. Reglas

El buscaminas (o *minesweeper*) se juega en un tablero cuadrado de $n \times n$ casilleros, el cual esconde un cierto número de minas. Inicialmente todos los casilleros están ocultos y el jugador debe ir descubriéndolos de a uno, evitando elegir uno que contenga una mina. Al descubrir un casillero se visualiza un número que indica la cantidad de minas que ese casillero tiene alrededor (ver figura 1). Si el casillero descubierto no tiene minas alrededor, se descubren automáticamente todos los casilleros adyacentes sin minas hasta encontrar, o bien el borde del tablero o bien un casillero con una mina alrededor.

Como ayuda-memoria, el jugador puede marcar un casillero como "posiblemente minado" utilizando una banderita roja. Esto no es validado por el juego, o sea, no pasa nada si el jugador marca como mina un casillero que no la tiene.

Si el jugador descubre una mina pierde la partida. El juego se gana al lograr descubrir todos los casilleros no minados.

Para comprender bien las reglas del juego, pueden abrir el Buscaminas en su sistema operativo favorito o jugarlo online aquí: `https://g.co/kgs/NdfZdM`

Figura 1: Buscaminas online, juego en desarrollo

4. Definiciones de tipos

Para nuestra especificación utilizaremos los siguientes renombres de tipos:

type $pos = \mathbb{Z} \times \mathbb{Z}$
: Identificador de una posición en el tablero (fila, columna). Ambos índices comienzan en cero. La fila avanza de arriba hacia abajo y la columna de izquierda a derecha (ver figura 2)

type $tablero = seq\langle seq\langle \text{Bool} \rangle \rangle$
: Matriz con valores booleanos indicando las posiciones de las minas. Cada posición contiene el valor *true* si hay una mina y el valor $false$ si no la hay. Cada elemento de la secuencia representa una fila, o sea que si t es de tipo tablero, $t[0]$ es la primera fila, $t[1]$ la segunda, etc. Las filas se cuentan de arriba hacia abajo. A su vez los elementos de cada fila se cuentan de izquierda a derecha, por lo que en $t[0][0]$ se encuentra la posición $(0, 0)$ y, en general, en $t[i][j]$ se encuentra la posición (i, j) (ver figura 2).

type $jugadas = seq\langle pos \times \mathbb{Z} \rangle$
: Secuencia de casillas jugadas. Incluye sólo las posiciones de las casillas descubiertas e indica, para una determinada posición, el número de minas adyacentes.

type $\text{banderitas} = seq\langle pos \rangle$
: Secuencia con las posiciones en las que el jugador puso una bandera porque considera que hay una mina (ayuda-memoria). El orden de los elementos de la secuencia no es importante.

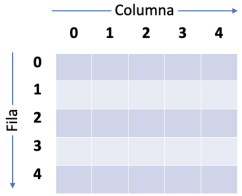

Figura 2: Posiciones

5. Ejercicios

5.1. Parte 1: Juego básico

Vamos a empezar a especificar las partes básicas del buscaminas, con algunas reglas simplificadas. En esta parte NO especificaremos el comportamiento de destapar automáticamente los casilleros adyacentes sin minas: sólo se descubrirá el casillero en la posición jugada. En caso de descubrir un casillero sin minas alrededor, solo aparecerá un 0 asociado a dicha posición. Se pide especificar los siguientes predicados, procedimientos y funciones auxiliares:

Ejercicio 1. aux minasAdyacentes(t: *tablero*, p: *pos*): $\mathbb{Z}$
Dado un tablero válido t y una posición válida p, esta función devuelve la cantidad de minas adyacentes a p, de acuerdo con las reglas del juego. Dos casilleros se consideran adyacentes si se tocan de forma vertical, horizontal o diagonal.

Ejercicio 2. pred juegoValido(t: *tablero*, j: *jugadas*)
Indica si el estado del juego representado por j es válido y se corresponde con la ubicación real de las minas representada por un tablero t. Sugerencia: escriban primero en castellano qué condiciones se requieren y recién después pásenlas al lenguaje de especificación.

Ejercicio 3. proc plantarBanderita(~~in n: $\mathbb{Z}$,~~ in t: *tablero*, in j: *jugadas*, in p: *pos*, inout b: *banderitas*)
El jugador marca la posición p con una banderita en el ayuda-memoria, indicando que en su opinión en esa posición hay una mina. ~~En este caso n es el tamaño del tablero.~~ Nótese que no se debe verificar la existencia real de una mina en esa posición (¡por eso es un ayuda-memoria!). Tenga en cuenta también que no se puede plantar una banderita en un casillero ya descubierto ni en un casillero que ya tiene una banderita.

Ejercicio 4. proc perdió(in t: *tablero*, in j: *jugadas*, out res: Bool)
Devuelve res = true si el jugador ha perdido el juego y res = false si el jugador no ha perdido.

Ejercicio 5. proc ganó(in t: *tablero*, in j: *jugadas*, out res: Bool)
Devuelve res = true si el jugador ha ganado el juego y res = false si el jugador no ha ganado.

Ejercicio 6. proc jugar(in t: *tablero*, in b: *banderitas*, in p: *pos*, inout j: *jugadas*)
A partir de un tablero válido y de un estado de juego válido, en el que el jugador aún no ganó ni perdió, este procedimiento modifica el estado del juego tras descubrir la posición p. No se puede jugar dos veces la misma posición ni se puede jugar una posición que tiene una banderita.

Sugerencia: piensen bien todas las condiciones que es necesario que se cumplan para que sea posible ejecutar un procedimiento. ¡No se puede "asumir" nada sobre los parámetros de entrada sin escribirlo en la precondición!

5.2. Parte 2: Despejar los vacíos

Vamos a intentar ahora modificar la función jugar para que si el jugador descubre un casillero sin ninguna mina alrededor se descubran automáticamente todos los casilleros en esa condición, hasta llegar a un casillero con número (ver figura 3). ¡Recuerden que en nuestro lenguaje de especificación no podemos usar recursión! Usaremos como ayuda el predicado del ejercicio que sigue, que también deberán especificar.

Ejercicio 7. pred caminoLibre(t: *tablero*, p_0: *pos*, p_1: *pos*)
Hay un camino entre p_0 y p_1 tal que ninguna posición intermedia del camino tiene minas alrededor. La posición p_0 no tiene minas alrededor pero la posición p_1 sí tiene al menos una. Se consideran posiciones adyacentes si comparten un vértice (o sea que también se consideran adyacentes si están en diagonal).

Ejercicio 8. proc jugarPlus(in t: *tablero*, in b: *banderitas*, in p: *pos*, inout j: *jugadas*)
Igual que el procedimiento jugar pero descubriendo automáticamente los casilleros sin minas adyacentes.

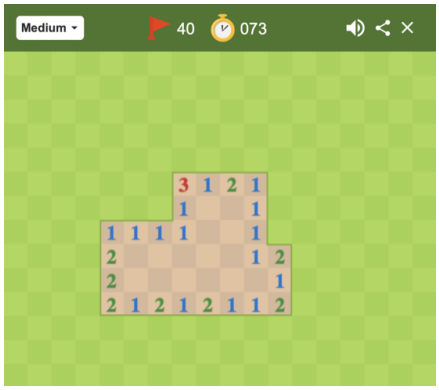

Figura 3: Al jugar una casilla sin minas alrededor, se descubren todas las casillas pegadas con la misma condición

5.3. Parte 3: Jugador automático

Si se entusiasmaron y se pusieron a *googlear*, seguro encontraron un montón de sitios con descripciones de patrones o secuencias de números que, de estar presentes en el juego, permiten realizar jugadas con la certeza de no perder. Intentaremos aquí especificar uno de ellos: el patrón $1-2-1$. Se puede observar que si en el tablero hay una secuencia de casilleros marcados con los números 1, 2 y 1 respectivamente (y alguna otra condición sobre los casilleros cercanos que deberán deducir), entonces hay un casillero que es posible destapar teniendo la absoluta seguridad de que no va a tener una mina.

Se pide especificar el siguiente procedimiento:

Ejercicio 9. proc sugerirAutomático121(~~in n: $\mathbb{Z}$,~~ in t: *tablero*, in b: *banderitas*, in j: *jugadas*, out p: *pos*)
Dado un estado de juego con un patrón $1-2-1$, devuelve en p una posición tal que, si se la juega, no se va a perder el juego. Nótese que este procedimiento ~~no toma un tablero como entrada porque~~ no tiene en cuenta la posición real de las minas sino sólo el estado del juego y el ayuda-memoria (que deben ser válidos).