Task 3 & 4: Minesweeper (Scheme+Haskell - 7+7 Points)

May 30, 2022

Implement a program to mark the number of mines directy adjacent (horizontally, vertically and diagonally) to squares on a Minesweeper field.
Example 1: Given this completed field as input (`'.'` denotes an empty field and `'*'` a mine)

```
.*.*.
..*..
..*..
.....
```

Your program should output the following

```
1*3*1
13*31
.2*2.
.111.
```

Hint

Start with a simpler task. Instead of solving the whole board, solve the problem for a specific square (x, y):

1. Get all the neighbours of (x, y) (staying in-bounds!).

$$\text{neighbours}(\text{board}, (x, y)) = [board_{(x-1,y-1)}, board_{(x-1,y)}, board_{(x-1,y+1)}, \ldots]$$

2. Count the number of mines among the neighbours.

3. Create a helper function capturing the printing rules: If the current square is a mine, return a star; otherwise, return the count of neighboring mines, or a dot if there are none. Then pass the surrounding mine-count and the square (x, y) as its arguments.

Now apply the procedure to every index. Use the same approach as in Task 2.

Task 3 — Scheme

Your file should have the extension `.rkt`. You may assume the input is rectangular and non-empty. Your program should read from standard input and write to standard output. Feel free to use the following skeleton.

```
-- for testing
(define test-board
  (map string->list (string-split ".*..\n..*.\n**..\n...*\n*...")))

-- for converting ints to chars.
(define (int->digit i) (integer->char (+ 48 i)))

(let* ((ss (port->lines))
       (ls (map string->list ss))
       (ln (sweep ls))              ; implement your function sweep
       (sn (map list->string ln)))
  (for ((l sn))
    (display l)
    (newline)))
```

The actual implementation is up to You.

Task 4 — Haskell

Your file should have the extension `.hs`. You may assume the input is rectangular and non-empty. Your program should read from standard input and write to standard output. Feel free to use the following skeleton.

```
-- for converting ints to chars
import Data.Char (intToDigit)

-- for testing
testBoard = lines ".*..\n..*.\n**..\n...*\n*..."

main = do
       str <- getContents
       let ls = lines str
       let sw = sweep ls -- implement your function sweep
       mapM_ putStrLn sw
```

The actual implementation is up to You.