Springer Nature Code Challenge

Below is a coding challenge that we would like you to solve. Please read through the description carefully and implement a solution for it. We don't want you to over-engineer the solution but be prepared to extend the functionality in the next step of the interview process. Finally, we ask you to submit a solution that you'd be happy to go live with and works "out of the box".

Please complete the challenge in your preferred language and specify which language you have used when your return the completed test.

Please create a local git repository and then commit to this as you work as Zip files tend to get blocked by our firewall. When you have finished please email it to hannah.sheppard@springernature.com. We will then review it within 7 days. Please do not make your solution public.

Good Luck!

Things We Value

- Working software!
- Tests.
- A working build.
- Small checkins with good comments.
- A simple readme with build and run instructions, and maybe talk about trade offs and design decisions you made.
- Simple code (but not necessarily easy!)
- The fewer libraries the better - we want to see your code. If you do use a library then explain why in the readme.
- We like functional constructs but also value good domain names and modelling.
- Evidence you have thought about edge cases and errors (either in code or the readme).

Enough talk, the Problem

You're given the task of writing a simple console version of a drawing program. The functionality of the program is quite limited but should be extensible. The program should work as follows:

1. create a new canvas.
2. start drawing on the canvas by issuing various commands.
3. quit.

The program should support the following commands:

Command	Description
`C w h`	Should create a new canvas of width w and height h.
`L x1 y1 x2 y2`	Should create a new line from `(x1,y1)` to `(x2,y2)`. Currently only horizontal or vertical lines are supported. Horizontal and vertical lines will be drawn using the `x` character.
`R x1 y1 x2 y2`	Should create a new rectangle, whose upper left corner is `(x1,y1)` and lower right corner is `(x2,y2)`. Horizontal and vertical lines will be drawn using the `x` character.
`B x y c`	Should fill the entire area connected to `(x,y)` with colour `'c'`. The behaviour of this is the same as that of the "bucket fill" tool in paint programs.
`Q`	Should quit the program.

Sample I/O

Below is a sample of the output your program should produce. User input is prefixed with `enter command:`.

```
enter command: C 20 4
----------------------
|                    |
|                    |
|                    |
|                    |
----------------------

enter command: L 1 2 6 2
----------------------
|                    |
|xxxxxx              |
|                    |
|                    |
----------------------

enter command: L 6 3 6 4
----------------------
|                    |
|xxxxxx              |
|     x              |
|     x              |
----------------------

enter command: R 16 1 20 3
----------------------
|               xxxxx|
|xxxxxx         x   x|
|     x         xxxxx|
|     x              |
----------------------

enter command: B 10 3 o
----------------------
|ooooooooooooooxxxxx|
|xxxxxxoooooooooox   x|
|     xooooooooxxxxx|
|     xooooooooooooo|
----------------------

enter command: Q
```