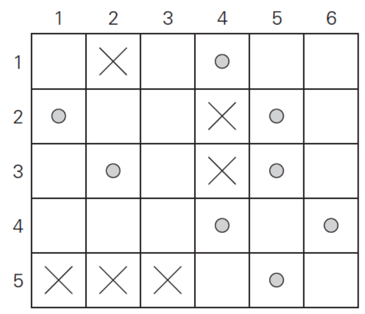
Eclipse, Kliezl P. Exercise 8.1 – 5

Q: How would you modify the dynamic programming algorithm for the coin-collecting problem if some cells on the board are inaccessible for the robot? Apply your algorithm to the board below, where the inaccessible cells are shown by X's. How many optimal paths are there for this board?



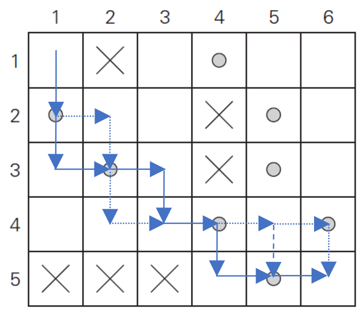
A:

```
ALGORITHM RobotCoinCollection(C[1..n, 1..m])
        //Applies dynamic programming to compute the largest number of coins a robot can collect on an
        n × m board by starting at (1, 1) and moving right and down from upper left to down right corner

        //Input: Matrix C[1..n, 1..m] whose elements are equal to 1 and 0 for cells with and without a coin,
        respectively, and equal to -1 for cells without access

        //Output: Largest number of coins the robot can bring to cell (n, m)

        $F[1, 1] \leftarrow C[1, 1]$
        for $j \leftarrow 2$ to $m$ do
                if $C[1][j] \neq -1$
                        $F[1, j] \leftarrow F[1, j-1] + C[1, j]$
                else $F[1, j] = -m$
        for $i \leftarrow 2$ to $n$ do
                $F[i, 1] \leftarrow F[i-1, 1] + C[i, 1]$
                for $j \leftarrow 2$ to $m$ do
                        if $C[i][j] \neq -1$
                                $F[i, j] \leftarrow max(F[i-1, j], F[i, j-1]) + C[i, j]$
                        else $F[i, j] = -m$
        return $F[n, m]$
```

In this example, the result will look like this:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 0 | -6 | -6 | -5 | -5 | -5 |
| 2 | 1 | 1 | 1 | -6 | -4 | -4 |
| 3 | 1 | 2 | 2 | -6 | -4 | -4 |
| 4 | 1 | 2 | 2 | 3 | 3 | 4 |
| 5 | -6 | -6 | -6 | 3 | 4 | 4 |

With this, there are 12 optimal paths for the board, which all collects 4 coins in total:


1
2
3
4
5
6
1
2
3
4
5