Shaan Mathur

Smallberg

CS32

April 26, 2016

Homework 2 Report

2.) Path of Stack Algorithm:

(6,4)

(6,3)

(6,5)

(7,5)

(8,5)

(8,6)

(8,7)

(8,8)

(7,8)

(6,6)

(5,4)

(4,4)

4.) Path of Queue Algorithm

(6,4)

(5,4)

(6,5)

(6,3)

(4,4)

(6,6)

(7,5)

(3,4)

(4,5)

(8,5)

(2,4)

(4,6)

The stack algorithm was implemented in such a way that it stacks first North, then East, South, and finally West. Because West is always placed last, the algorithm prefers to search deep West as much as it possibly can. If it cannot move West, it will then move South a space and see if it can move West again. If South fails, it will then try East (upon which it will try West once more). The last direction it checks is North. This is all due to the nature of stacks to check the last point pushed in. This algorithm chooses to put West last, hence this behavior.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| X | X | X | X | X | X | X | X | X | X |
| X |  |  |  |  |  |  |  |  | X |
| X |  | X |  |  |  |  | X |  | X |
| X | X | 🡨 | 🡨 | **Start** |  |  |  |  | X |
| X | 🡨 | 🡫 |  |  |  | X |  |  | X |
| X | 🡫 | 🡪 | 🡪 |  | **End** | 🡨 | 🡨 | 🡩 | X |
| X | 🡫 | X | 🡫 | 🡪 | 🡪 | 🡪 | 🡪 | 🡪 | X |
| X | X | X | X | X | X | X | X | X | X |

On the other hand, the queue algorithm takes a different approach. It checks North, East, South, and then West. Then at these new points, it checks North, East, South, West (if it can). In other words, it branches out rather evenly (producing an algorithm pattern like is that of a fractal). This queue evenly checks all points, because any new points it adds to the queue will be checked last (so it will check all old points before checking the newest point).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| X | X | X | X | X | X | X | X | X | X |
| X |  |  | 🡨 | 🡩 | 🡪 |  |  |  | X |
| X |  | X | 🡨 | 🡩 | 🡪 | 🡩 | X |  | X |
| X | **End** | 🡨 | 🡨 | **Start** | 🡪 | 🡪 | 🡪 |  | X |
| X |  | 🡫 | 🡨 | 🡫 | 🡪 | X |  |  | X |
| X |  |  | 🡨 | 🡫 | 🡪 |  |  |  | X |
| X |  | X |  | 🡫 |  |  |  |  | X |
| X | X | X | X | X | X | X | X | X | X |