p1_search_q2_bfs

Question 2 (3 points): Breadth First Search

Implement the breadth-first search (BFS) algorithm in the `breadthFirstSearch` function in `search.py`. Again, write a graph search algorithm that avoids expanding any already visited states. Test your code the same way you did for depth-first search.

```
python pacman.py -l mediumMaze -p SearchAgent -a fn=bfs
```

```
python pacman.py -l bigMaze -p SearchAgent -a fn=bfs -z .5
```

Does BFS find a least cost solution? If not, check your implementation.

Hint: If Pacman moves too slowly for you, try the option `--frameTime 0`.

Note: If you've written your search code generically, your code should work equally well for the eight-puzzle search problem without any changes.


```
python eightpuzzle.py
```