Assignment 7 Problem 1

The algorithm should do the following:

Set a pointer point to the location in the square grid where we currently are.

Then for each ? in the string s, we try all possibilities (L, R, U, or D), but we do not try the one that is the opposite of the previous bit (i.e. if s[i] = L and s[i+1] = ?, then we try s[i+1] = L, U and D) since we are seeking a Hamiltonian path.

Note that if the first ? is the first entry of the string s, then we try all four directions.

By doing this, we constructs $4 \times 3^{k-1} \in O(3^k)$ different strings s. And each of string s is of length $n^2 - 1 \in O(n^2)$.

To know whether the Hamiltonian path exist, the algorithm should check all strings that we constructs. By looking through the entire string and change the pointer in the corresponding way, we can construct a path that represented by the string. If, at any time, the pointer is out of range (points outside of the grid), then such string is not a valid Hamiltonian path, reset the pointer, and check next string. Also, if the pointer successfully goes over the whole string, but it does not point to the right-bottom position, the string is also not a valid Hamiltonian path, rest and check next. Otherwise, the string represents a Hamiltonian path. If algorithm gets a Hamiltonian path, then returns True. If no Hamiltonian path found after checking all strings, then returns False.

For each string, the time of checking it is depending on the length of the string which is $O(n^2)$.

Pseudo code:

```
$chekcing(s)$
$s$: $the\ string\ that\ may\ represent\ a\ Hamiltonian\ path$
    $ptr = location\ of\ top - left$
    $for\ i = 1\ to\ n^2 - 1\ do$
        $move\ ptr\ according\ to\ s[i]$
        $if\ ptr\ out\ of\ bound$
            $return\ False$
    $if\ ptr\ points\ to\ the\ bottom - right$
        $return\ True$
    $return\ False$
```

```
$find\_path(n, s)$
    $construct\ all\ valid\ combinations\ of\ s\ with\ \{L, R, U, D\}$
    $while\ not\ all\ combinations\ tried$
        $if(checking(current\_combination))$
            $return\ True$
    $return\ False$
```

Runtime: Total time constructing combinations $O(3^k)$, number of constructed combinations $O(3^k)$, and for each checking $O(n^2)$. The total runtime is $O(n^2 3^k + 3^k) = O(n^2 3^k)$