Day 10

Part 1

You all arrive at a Lava Production Facility on a floating island in the sky. As the others begin to search the massive industrial complex, you feel a small nose boop your leg and look down to discover a reindeer wearing a hard hat.

The reindeer is holding a book titled "Lava Island Hiking Guide". However, when you open the book, you discover that most of it seems to have been scorched by lava! As you're about to ask how you can help, the reindeer brings you a blank topographic map of the surrounding area (your puzzle input) and looks up at you excitedly.

Perhaps you can help fill in the missing hiking trails?

The topographic map indicates the height at each position using a scale from 0 (lowest) to 9 (highest). For example:

```
0123
1234
8765
9876
```

Based on un-scorched scraps of the book, you determine that a good hiking trail is as long as possible and has an even, gradual, uphill slope. For all practical purposes, this means that a hiking trail is any path that starts at height 0, ends at height 9, and always increases by a height of exactly 1 at each step. Hiking trails never include diagonal steps - only up, down, left, or right (from the perspective of the map).

You look up from the map and notice that the reindeer has helpfully begun to construct a small pile of pencils, markers, rulers, compasses, stickers, and other equipment you might need to update the map with hiking trails.

A trailhead is any position that starts one or more hiking trails - here, these positions will always have height 0. Assembling more fragments of pages, you establish that a trailhead's score is the number of 9-height positions reachable from that trailhead via a hiking trail. In the above example, the single trailhead in the top left corner has a score of 1 because it can reach a single 9 (the one in the bottom left).

This trailhead has a score of 2:

```
...0...
...1...
...2...
6543456
7.....7
8.....8
9.....9
```

(The positions marked . are impassable tiles to simplify these examples; they do not appear on your actual topographic map.)

This trailhead has a score of 4 because every 9 is reachable via a hiking trail except the one immediately to the left of the trailhead:

```
..90..9
...1.98
...2..7
6543456
765.987
876....
987....
```

This topographic map contains two trailheads; the trailhead at the top has a score of 1, while the trailhead at the bottom has a score of 2:

```
10..9..
2...8..
3...7..
4567654
...8..3
...9..2
.....01
```

Here's a larger example:

```
89010123
78121874
87430965
96549874
45678903
32019012
01329801
10456732
```

This larger example has 9 trailheads. Considering the trailheads in reading order, they have scores of 5, 6, 5, 3, 1, 3, 5, 3, and 5. Adding these scores together, the sum of the scores of all trailheads is `36`.

The reindeer gleefully carries over a protractor and adds it to the pile. What is the sum of the scores of all trailheads on your topographic map?

In [1]:
```python
from pathlib import Path

import networkx as nx
import numpy as np
```

In [2]:
```python
test_path = Path("data/day10_test.txt")
data_path = Path("data/day10_data.txt")

def load_data(fpath: Path) -> np.array:
    """Returns array representation of topographic map."""
    topo = []

    with fpath.open() as ifh:
        for row in [list(_.strip()) for _ in ifh.readlines()]:
            topo.append([int(_) for _ in row])

    return np.array(topo)
```

```python
def add_topog_edges(topog: nx.DiGraph, fromloc: tuple[int, int], toloc: tuple[int, in
    """Adds directed edge to a topographical digraph.

    Edges are weighted by the difference in height attribute between
    the two connected nodes.
    """
    step = topog.nodes[toloc]["height"] - topog.nodes[fromloc]["height"]
    topog.add_edge(fromloc, toloc, step=step)
    return topog



def topo_to_graph(topo: np.array) -> nx.DiGraph:
    """Returns DiGraph representation of topographical array.

    Nodes are indexed by array location and carry a height attribute.
    Directed edges are added between nodes with a step attribute that
    represents the height change when travelling between edges.
    """
    nrows, ncols = topo.shape

    topog = nx.DiGraph()

    # Add nodes to graph
    for row in range(nrows):
        for col in range(ncols):
            topog.add_node((row, col), height=int(topo[row][col]))

    # Add edges to graph
    for row in range(nrows):
        for col in range(ncols):
            if row > 0:  # add N
                topog = add_topog_edges(topog, (row, col), (row-1, col))
            if row < nrows-1:  # add S
                topog = add_topog_edges(topog, (row, col), (row+1, col))
            if col > 0:  # add W
                topog = add_topog_edges(topog, (row, col), (row, col-1))
            if col < nrows-1:  # add E
                topog = add_topog_edges(topog, (row, col), (row, col+1))

    return topog


def isolate_trails(topog: nx.DiGraph) -> nx.DiGraph:
    """Returns a topographical graph having all non-trail edges removed."""
    # Remove all edges with a weight that is not 1
    for edge in list(topog.edges):
        uidx, vidx = edge
        if topog.edges[*edge]["step"] != 1:  # Not a valid trail edge
            topog.remove_edge(*edge)

    return topog


def find_trailhead_nodes(topog: nx.Graph) -> list[tuple[int, int]]:
    """Returns list of trailhead nodes."""
    return [_ for _ in topog.nodes if topog.nodes[_]["height"] == 0]


def find_peaks(topog: nx.Graph) -> list[tuple[int, int]]:
    """Returns list of topographical peaks."""
    return [_ for _ in topog.nodes if topog.nodes[_]["height"] == 9]


def score_trailheads(topog: nx.DiGraph, trailheads: list[tuple[int, int]], peaks: lis
    """Returns list of scores for each trailhead."""
    scores = []

    for thead in trailheads:
        peakcount = 0  # Count of reachable distinct peaks
        for peak in peaks:
            if nx.has_path(topog, thead, peak):  # Path exists
                peakcount += 1
        scores.append(peakcount)

    return scores


topo = load_data(test_path)
topog = topo_to_graph(topo)
trails = isolate_trails(topog)
trailheads = find_trailhead_nodes(trails)
peaks = find_peaks(trails)
sum(score_trailheads(trails, trailheads, peaks))
```

Out[2]: 36

In [3]:
```
topo = load_data(data_path)
topog = topo_to_graph(topo)
trails = isolate_trails(topog)
trailheads = find_trailhead_nodes(trails)
peaks = find_peaks(trails)
sum(score_trailheads(trails, trailheads, peaks))
```

Out[3]: 587

Part 2

The reindeer spends a few minutes reviewing your hiking trail map before realizing something, disappearing for a few minutes, and finally returning with yet another slightly-charred piece of paper.

The paper describes a second way to measure a trailhead called its rating. A trailhead's rating is the number of distinct hiking trails which begin at that trailhead. For example:

```
.....0.
..4321.
..5..2.
..6543.
..7..4.
..8765.
..9....
```

The above map has a single trailhead; its rating is 3 because there are exactly three distinct hiking trails which begin at that position:

```
.....0.   .....0.   .....0.
..4321.   .....1.   .....1.
..5....   .....2.   .....2.
..6....   ..6543.   .....3.
..7....   ..7....   .....4.
..8....   ..8....   ..8765.
..9....   ..9....   ..9....
```

Here is a map containing a single trailhead with rating 13:

```
..90..9
...1.98
...2..7
6543456
765.987
876....
987....
```

This map contains a single trailhead with rating 227 (because there are 121 distinct hiking trails that lead to the 9 on the right edge and 106 that lead to the 9 on the bottom edge):

```
012345
123456
234567
345678
4.6789
56789.
```

Here's the larger example from before:

```
89010123
78121874
87430965
96549874
45678903
32019012
01329801
10456732
```

Considering its trailheads in reading order, they have ratings of 20, 24, 10, 4, 1, 4, 5, 8, and 5. The sum of all trailhead ratings in this larger example topographic map is `81`.

You're not sure how, but the reindeer seems to have crafted some tiny flags out of toothpicks and bits of paper and is using them to mark trailheads on your topographic map. What is the sum of the ratings of all trailheads?

In [4]:

```python
def rate_trailheads(topog: nx.DiGraph, trailheads: list[tuple[int, int]], peaks: list
    """Returns list of ratings for each trailhead."""
    ratings = []

    for thead in trailheads:
        paths = 0  # Count of paths from trailhead to peaks
        for peak in peaks:
            paths += len(list(nx.all_simple_paths(topog, thead, peak)))
        ratings.append(paths)

    return ratings


topo = load_data(test_path)
topog = topo_to_graph(topo)
trails = isolate_trails(topog)
trailheads = find_trailhead_nodes(trails)
peaks = find_peaks(trails)
sum(rate_trailheads(trails, trailheads, peaks))
```

Out[4]:

```
81
```

In [5]:

```python
topo = load_data(data_path)
topog = topo_to_graph(topo)
```

```
trails = isolate_trails(topog)
trailheads = find_trailhead_nodes(trails)
peaks = find_peaks(trails)
sum(rate_trailheads(trails, trailheads, peaks))
```

Out[5]:

```
1340
```

In [ ]: