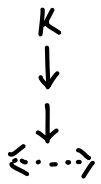
- A Knight in chess can move any vertical/horizontal direction two spaces, followed by one space perpendicular to its 2 space motion



- Translate this to a phone pad, given the length of digit is n ($1 \leq n \leq 5'000$) how many unique digits can it produce?

$n = 3$

1	2	3
4	5	6
7	8	9
-	0	-

1	2	3
4	5	6
7	8	9
	0	

Starting @ 1: 1 8 1

1 8 3

1 6 7

1 6 1

1 6 1 6, 1618, 1672, 1676

1 8 1 8, 1816, 1838, 1834

$\text{unique}_n = 2^{n-1}$

For every move after 1, it can choose 2 paths

1, 2, 4, 8, 16

Starting @ 2:

$n = 1$, $l = 1$

$n = 2$, $l = 4$

2 7 2, 292, 276, 294

$n = 3$, $l = 10$

2 7 2 7, 2729, 2761, 2767, 2760

2 9 2 9, 2927, 2943, 2949, 2940

$n = 4$

There might be a solution involving the summation of equations too hard to figure out.

Realistic: DFS

Ex. 1 → (6, 8)
2 → (9, 7)
3 → (4, 8)
4 → (0, 3, 9)
5 → ( )
6 → (0, 1, 7)
7 → (6, 2)
8 → (1, 3)
9 → (4, 2)
0 → (4, 6)

```
f(n: int)
  for i = 0 → 10
    ans = unique(n-1, i)

unique(n_left: int, value: int, &ans: int)
  if n_left < 1
    return 1
  for values → m[value]
    X for Key → values
        ~~return~~ ans +=  unique(n-1, Key)
  return ans
```

This is a brute force but can be improved by "caching" values we have seen before.

cache : map<pair<int, int>, long long>

Instead of ans+= ...

```
cache[pair(n-1, Key)] = unique...
ans += cache[pair(n-1, key)]
```

Check if cache contains a value b4 calculating.

This solution greatly speeds up algorithm without much manipulation to the function.