20.10.12 동아리 풀이

선린인터넷고등학교 소프트웨어과
30610 나정휘
https://JusticeHui.github.io

문제 목록

- A. Bombs In My Deck - 2020 KAIST MOCK B번
- B. 우체국
- C. 영재의 산책 - 2020 인하대 교내대회 J번
- D. 약수 게임
- E. 소방서의 고민
- F. 캔디 분배 - ICPC 독일 리저널 2012 C번
- G. Remote Control - 2020 KAIST MOCK I번
- H. 성벽 - 2015 일본 정보 올림피아드 5번

Bombs In My Deck (1)

- $T := \left\lfloor \frac{C+4}{5} \right\rfloor = \left\lceil \frac{C}{5} \right\rceil$
- 1 - (폭탄을 T번 연속으로 뽑을 확률) 을 구하는 문제

- 단순 구현 문제

- http://boj.kr/25f1445a26144dab8328a88cd155d094

우체국 (1)

- ***Theorem.*** 사람들의 거주지를 오름차순으로 정렬했을 때, 중앙값이 문제의 정답이 된다.
- ***Proof.*** 알아서 하자.

- 이것도 단순 구현 문제
- 정렬하고 앞에서부터 보면서 처음으로 절반 이상이 되는 지점

- http://boj.kr/1d24464728f845a1b686b45f6abbb313

영재의 산책 (1)

- 수열 $A_n = x^n \ mod \ 10$ 의 주기는 1, 2, 4 중 하나다.
  - 주기가 4라고 생각해도 된다.

- 4방향으로 이동하는 횟수를 각각 구한 뒤 ($\left\lfloor \frac{T}{4} \right\rfloor, \left\lceil \frac{T}{4} \right\rceil$ 중 하나다.)
- 정답을 상수 시간에 계산하면 된다.
  - 처음에는 v만큼 이동하는 것에 주의

- http://boj.kr/8d2e236051314c13b7f1206600985769

약수 게임 (1)

- N이 합성수인 경우에는 구사과가 이길 수 있다.
  - 구사과가 소수로 만든다.
  - 큐브러버가 1로 만든다. (소수이기 때문에 1로 만들 수 밖에 없다.)
  - 구사과가 이긴다.
- N이 소수인 경우에는 큐브러버가 이긴다.
  - 구사과가 1로 만든다. (소수이기 때문에 1로 만들 수 밖에 없다.)
  - 큐브러버가 이긴다.
- O(√N)에 소인수분해를 하면 된다.
- http://boj.kr/9d0cdf65e5a24e70a4cb59fc8d14d08d

소방서의 고민 (1)

- Exchange Argument 를 쓰면 된다.

- 어떤 두 화재 $(a_1, b_1), (a_2, b_2)$가 있다고 하자.
  - $b_1 a_2 < a_1 b_2$인 경우에는 (a_1, b_1)를 먼저 처리하는 것이 이득이다.
  - $b_1 a_2 > a_1 b_2$인 경우에는 (a_2, b_2)를 먼저 처리하는 것이 이득이다.

소방서의 고민 (2)

- ***Proof.*** 식 정리를 잘 하면 Exchange Argument를 통해 최적임을 보일 수 있다.

- http://boj.kr/fa3af9c0dd8f4006a2d37dbc6155738a

캔디 분배 (1)

- 사탕이 C개 들어있는 봉지를 X개 사는 것이 정답이라고 하자.
- CX를 K로 나눈 나머지가 1이 되어야 한다.

- $CX \equiv 1 \ (mod\ K)$
- $X \equiv C^{-1} \ (mod\ K)$, 즉 C의 모듈러 인버스를 구하면 된다.

캔디 분배 (2)

- 예외 처리 해야하는 것
  - $\gcd(C, K) \neq 1$이면 모듈러 인버스는 존재하지 않는다.
  - C = 1인 경우
  - K = 1인 경우
  - C = 1이면서 K = 1인 경우
  - 귀찮다!

- http://boj.kr/b5f1c1f4ebb84d8b956a3e8c78f1fc92

Remote Control (1)

- 모든 좌표(쿼리)를 입력 받은 다음, 한 번에 처리하자.

- 각 턴마다 차 Q대를 전부 움직이는 것은 미친 짓 같으니
- 벽을 움직인다고 생각하자.

Remote Control (2)

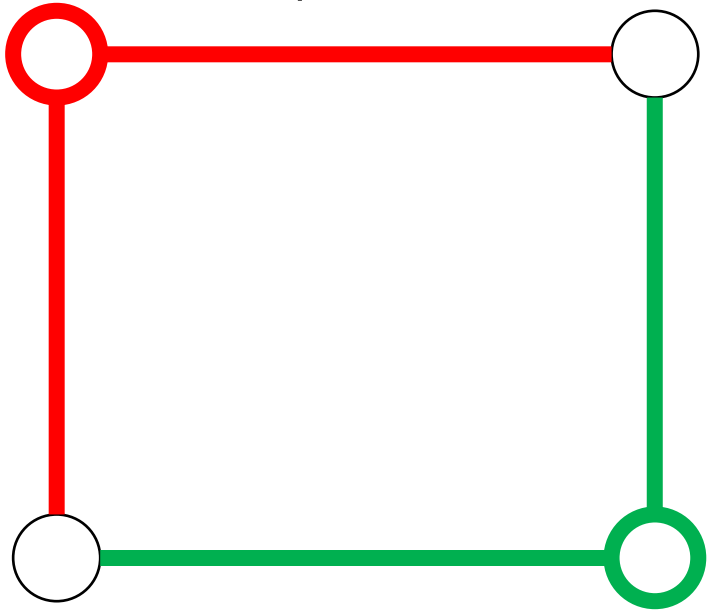
- 만약 어떤 차가 벽이랑 충돌하면 그 행동은 캔슬된다.
  - 그 차도 벽이 이동하는 방향으로 같이 밀어주면 된다!

- 어떤 두 차가 동일한 칸에 속한다면, 그 시점 이후에도 계속 같은 칸에 속한다.
- Union Find로 묶어서 관리하면 된다.

- http://boj.kr/4b01d5a35b364a3c8b026c438941be19

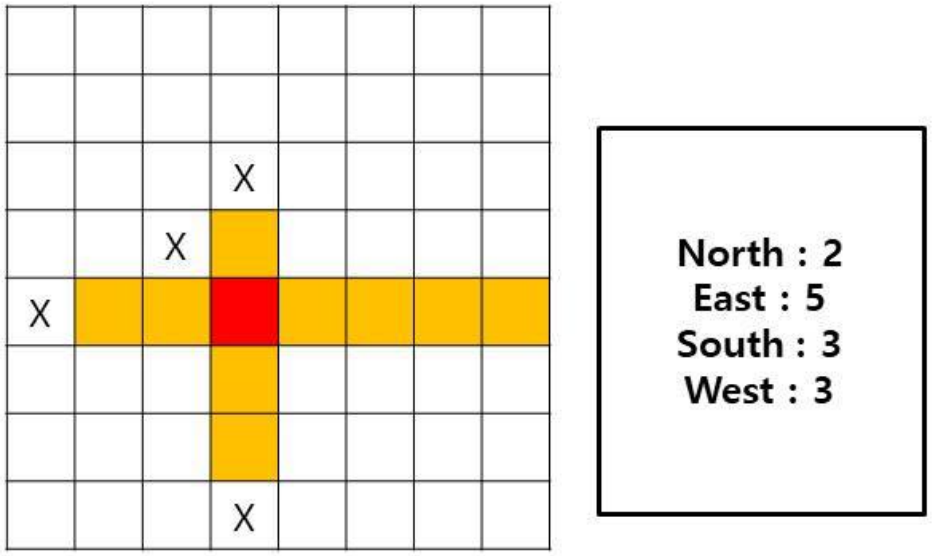
성벽 (1)

- 성벽을 2부분으로 분할하자.
- 북서쪽에서 뻗어나가는 성벽과 남동쪽에서 뻗어나가는 성벽으로 나눠서 생각할 것이다.
- 북서-동남을 연결하는 대각선은 H+W-1개 있고, 각 대각선에 대해 각각 정답을 구해주면 된다.



성벽 (2)

- 각 점에서 4방향으로 뻗어나갈 수 있는 최대 길이를 구하자.
- DP로 O(HW)에 구할 수 있다.



- 이미지 출처 : 페이스북 "탐레프 하는 백준"

성벽 (3)

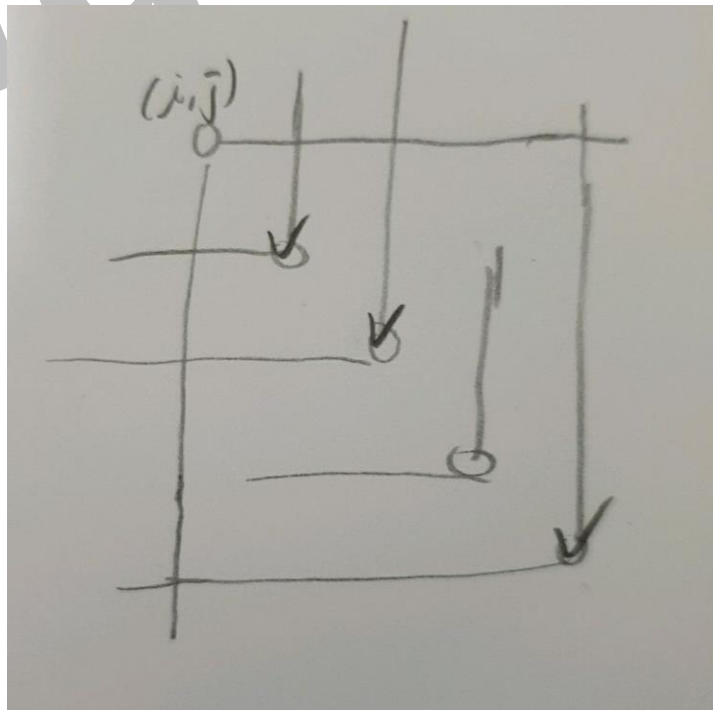
- 어떤 대각선에 대해...

- 일단 대각선 위에 있는 모든 점 (i, j)에 대해서
  - 그 점에서 시작하는 남동쪽 성벽이 어디까지 뻗을 수 있는지 구하자.
  - i - min{ west(i, j), north(i, j) } + 1
  - (i-min{ west(i, j), north(i, j) }+1, i) pair를 저장한 배열을 v라고 할 때
  - v를 정렬하자.

성벽 (4)

- 대각선의 위쪽부터 훑으면서
- 현재 점 (i, j)에서 뻗어나가는 북서쪽 성벽과 매칭 가능한 남동쪽 성벽의 개수를 구하면 된다.

- inversion counting 느낌으로 해주면 된다.



성벽 (5)

```
for(int i=si, j=sj; i<=n && j<=m; i++, j++){
    range.emplace_back(i-min(north[i][j], west[i][j])+1, i);
}
sort(all(range));
for(int i=si, j=sj; i<=n && j<=m; i++, j++){
    while(pv < range.size() && range[pv].x <= i) update(range[pv++].y, 1);
    ans += query(i+1-1, i+min(south[i][j], east[i][j])-1);
}
```

- http://boj.kr/8b5b2406ed854ac49d49d6e2b8da0238