목차

문제		의도한 난이도	출제자
A	YCPC 점수	Beginner	박경욱sk091204091204
B	연대 다음 고대	Beginner	강호현annieho
C	점 찾기	Easy	박경욱sk091204091204
D	욘세이대학교	Medium	김형진plast
E	Yonsei TOTO 2	Medium	오윤석oh040411
F	호현이와 파이썬	Hard	이재열ConU
G	멀티팩토리얼	Hard	오윤석oh040411
H	매드 맥스	Hard	오윤석oh040411
I	아름다운 수열 만들기	Challenging	김형진plast
J	Bracket Sequence Endgame	Challenging	김태윤ystaeyoon113
K	초콜릿 먹기	Plast	김형진plast
L	초콜릿 놓기	Plast	김형진plast

출제진 소개

- 강호현(annieho) - 연세대학교 도시공학과
- 김태윤(ystaeyoon113) - 연세대학교 컴퓨터과학과
- 김형진(plast) - 연세대학교 컴퓨터과학과
- 박경욱(sk091204091204) - 연세대학교 인공지능학과
- 오윤석(oh040411) - 연세대학교 물리학과
- 이재열(ConU) - 연세대학교 컴퓨터과학과

검수진 소개

- 김건우 (coconut99) - 삼성전자, 연세대학교 컴퓨터과학과
- pyb1031 - 비공개
- 신용명 (tlsdydaud1) - 한양대학교 컴퓨터소프트웨어학부
- 이국렬 (lky7674) - 삼성전자, 연세대학교 컴퓨터과학과
- 이하람 (rammma) - 연세대학교 컴퓨터과학과
- 장래오 (leo020630) - 포항공과대학교 (명예 연세인)
- 정용진 (playsworld16) - 연세대학교 컴퓨터과학과

A. YCPC 점수

`mathematics`
출제진 의도 – **Beginner**

- ✓ 제출 158번, 정답 127명 (정답률 81.646%)
- ✓ 처음 푼 사람: **cohenpf**, 0분
- ✓ 출제자: **박경욱(sk091204091204)**

A. YCPC 점수

- ✓ 문자열 YCPC를 만들기 위해 문자 Y, C, P가 각각 1개, 2개, 1개가 필요합니다.
- ✓ 따라서, $\min(y, c/2, p)$ 를 계산하면 됩니다.

B. 연대 다음 고대

`string`, `implementation`
출제진 의도 – **Beginner**

- ✓ 제출 130번, 정답 100명 (정답률 77.692%)
- ✓ 처음 푼 사람: **leinad2**, 1분
- ✓ 출제자: **강호현(annieho)**

B. 연대 다음 고대

- ✓ Korea가 등장하기 전 Yonsei가 등장하는 경우 Yonsei Won!을, 반대의 경우 Yonsei Lost...를 출력합니다.

C. 점 찾기

`parity`, `case work`, `mathematics`
출제진 의도 – **Easy**

- ✓ 제출 147번, 정답 40명 (정답률 27.211%)
- ✓ 처음 푼 사람: **nflight11**, 2분
- ✓ 출제자: **박경욱(sk091204091204)**

C. 점 찾기

- ✓ 이 문제를 풀 수 있는 방법은 매우 다양합니다.
- ✓ 먼저, $x_a + y_a$와 $x_b + y_b$의 홀짝성이 동일하면 문제 조건을 만족하는 점 p가 존재하고, 다르면 존재하지 않습니다.
- ✓ $\Delta x = |x_a - x_b|, \Delta y = |y_a - y_b|$라고 할 때, 1) $\Delta x \equiv \Delta y \equiv 0 \mod 2$인 경우와 2) $\Delta x \equiv \Delta y \equiv 1 \mod 2$인 경우로 나눌 수 있습니다.

C. 점 찾기

- ✓ Case 1: $\Delta x \equiv \Delta y \equiv 0 \mod 2$
- ✓ $p = (\frac{x_a + x_b}{2}, \frac{y_a + y_b}{2})$ 가 문제 조건을 만족합니다.
- ✓ Case 2: $\Delta x \equiv \Delta y \equiv 1 \mod 2$
- ✓ 일반성을 잃지 않고 $x_a < x_b$ 라고 합시다.
  - Case 2-1: $y_a < y_b$
  - $p = (\frac{x_a + x_b + 1}{2}, \frac{y_a + y_b - 1}{2})$ 가 문제 조건을 만족합니다.
  - Case 2-2: $y_a > y_b$
  - $p = (\frac{x_a + x_b - 1}{2}, \frac{y_a + y_b - 1}{2})$ 가 문제 조건을 만족합니다.

D. 욘세이대학교

`greedy, constructive`
출제진 의도 – **Medium**

- ✓ 제출 63번, 정답 27명 (정답률 42.857%)
- ✓ 처음 푼 사람: **urd05**, 12분
- ✓ 출제자: **김형진(plast)**

D. 욘세이대학교

- 단어를 가장 짧게 만들고 싶기 때문에, 주어진 단어에서 최소한의 알파벳을 추가해 멋이 있는 단어를 만들어야 합니다.
- 단어의 X, Y 가 $|X - Y| \leq 1$ 을 만족하면 이미 멋있는 단어입니다.
- 아니라면, 현재는 $|X - Y| > 1$ 을 만족합니다. 일반성을 잃지 않고 $X > Y$ 라고 합시다.
- 우리는 최소한 $X - Y - 1$ 개의 알파벳은 추가해야 합니다. 알파벳을 추가할 때마다 Y 가 최대 1 증가하기 때문입니다.

D. 욘세이대학교

- 그래서, 현재 상태에서 감소하는 $X - Y - 1$ 개의 알파벳을 사용할 수 있다면 정답은 $X - Y - 1$ 개를 사용하는 것이 최소입니다.
- 예를 들어, AEGZ $(X = 3, Y = 0)$ 은 Z 이하의 감소하는 2개의 알파벳 C, B를 사용함으로써 AEGYCB $(X = 3, Y = 2)$ 라는 멋있는 단어를 만들 수 있게 됩니다.
- 그러나 우리가 $X - Y - 1$ 개의 알파벳을 사용할 수 없을 때도 있습니다. ABDE $(X = 3, Y = 0)$ 은 E 이하의 알파벳이 1개밖에 남지 않았으므로 항상 Y 만을 증가시키는 위의 방법을 사용할 수 없습니다.

- 따라서 이 경우에는, 어쩔 수 없이 X 를 1 증가시키게 되어, Y 는 최소 $X - Y$ 번 이상 증가시켜야 합니다. 따라서 전체는 최소 $X - Y + 1$ 번의 연산을 추가해야 합니다.
- 이때 현재까지 등장하지 않은 가장 큰 알파벳을 계속 추가하는 전략으로 $X - Y + 1$ 이라는 하한을 만족시킬 수 있습니다.
- 가령 ABC $(X = 2, Y = 0)$ 라는 단어는, 각각 Z, Y, X 를 추가하는 방법으로 $(X = 3, Y = 2)$ 를 만족시킬 수 있습니다.
- 이 전략을 그리디하게 적용해도 되고, 추가되는 알파벳이 5 개 이하라는 것을 발견하여 백트래킹 등의 방법으로 완전탐색하는 방법도 있습니다.

E. Yonsei TOTO 2

`greedy`, `sorting`, `linearity of expectation`

출제진 의도 – **Medium**

- 제출 28번, 정답 23명 (정답률 82.143%)
- 처음 푼 사람: **leinad2**, 17분
- 출제자: **오윤석(oh040411)**

E. Yonsei TOTO 2

- i 번째 과목에서 얻는 만족도를 b_i, 수강신청 성공 확률을 P_i 라고 합시다. 문제의 목표는 $\mathbb{E}\left(\sum_{i=1}^{N} b_i\right)$ 를 최대화하는 것입니다.
- 기댓값의 선형성에 의해, $\mathbb{E}\left(\sum_{i=1}^{N} b_i\right) = \sum_{i=1}^{N} \mathbb{E}(b_i) = \sum_{i=1}^{N} P_i B_i$ 입니다.
- 한편, $X_i > A_i$ 마일리지를 베팅한 경우는 $X_i = A_i$ 마일리지를 베팅했을 때의 결과와 차이가 없으므로, A_i 를 초과하여 베팅하면 기댓값의 이득을 볼 수 없습니다. 따라서 $X_i > A_i$ 인 경우는 배제할 수 있고, $P_i = X_i / A_i$ 로 생각해도 됩니다.

E. Yonsei TOTO 2

- ✓ 따라서 $\sum_{i=1}^{N} \frac{B_i}{A_i} X_i$ 를 최대화하면 됩니다.
- ✓ 각 마일리지 한 표는 기댓값에 B_i/A_i 만큼 기여하므로, B_i/A_i 가 큰 과목부터 베팅하는 것이 이득입니다.
- ✓ 과목을 B_i/A_i 의 내림차순으로 정렬한 후, 총 S 마일리지가 전부 소진될 때까지 순서대로 최대 $\min(M, A_i)$ 마일리지를 베팅하면 됩니다. 만약 모든 과목에 베팅했는데 마일리지가 남았다면 그대로 둬도 괜찮습니다.
- ✓ B_i/A_i 의 분모는 $1\,000$ 이하이므로, 서로 다른 두 분수의 차이는 10^{-6} 이상입니다. 따라서 double 자료형 등을 사용해도 실수 오차로부터 안전합니다.
- ✓ 시간 복잡도는 정렬에 필요한 $\mathcal{O}(N \log N)$ 이며, 마일리지를 1 씩 베팅하는 방식으로 구현한 $\mathcal{O}(N \log N + NM)$ 풀이도 통과할 수 있습니다.

F. 호현이와 파이썬

`graph`, `eulerian path`
출제진 의도 – **Hard**

- ✓ 제출 93번, 정답 28명 (정답률 30.108%)
- ✓ 처음 푼 사람: **ychangseok**, 4분
- ✓ 출제자: **이재열 (ConU)**

F. 호현이와 파이썬

- ✓ 그래프 $G = (V, E)$ 에서 V 의 각 정점은 변수 하나에 해당하며, E 의 u 번 정점과 v 번 정점을 잇는 간선이 존재한다면 u 번째 변수와 v 번째 변수의 값이 다름을 의미한다고 합시다.
- ✓ 이때 문제의 조건을 만족시키기 위해서는 G 가 완전그래프여야 합니다. 즉, 어떤 두 점을 골라도 해당 두 점을 잇는 간선이 존재해야 합니다.
- ✓ 이 관점으로 보면, 하나의 != 연산자는 그래프상의 간선 하나에 해당합니다.
- ✓ 따라서 이 문제는 한붓그리기처럼 특정 정점에서 '출발'해 간선을 '타고 가는' 행위로 완전그래프를 '그리는' 문제가 됩니다.
- ✓ 설명의 편의를 위해, 완전그래프가 주어졌을 때 특정 정점에서 출발해 간선을 타고 가는 행위로 간선을 전부 지우는 문제라고 생각하겠습니다.

- ✓ != 연산자만 사용할 수 있기에, 한붓그리기 상황과 마찬가지로 붓을 뗄 수 없습니다.
- ✓ 하지만 한붓그리기 상황과는 다르게 같은 간선을 여러 번 지나갈 수는 있습니다. (즉, 이미 간선이 지워졌어도 지나갈 수는 있음)
- ✓ 완전그래프를 그리기 위해서 완전그래프의 간선 개수인 $\frac{N(N-1)}{2}$ 번은 != 연산자를 사용해야 합니다.
- ✓ 결국 실제로 구해야 하는 값은 같은 간선을 여러 번 지나가는 일을 최소한으로 줄였을 때 몇 번 지나가야 하는지입니다.

F. 호현이와 파이썬

- ✓ 그래프에서 정점의 차수란 해당 정점과 간선으로 바로 연결된 정점의 개수를 의미합니다.
- ✓ 홀수 차수 정점의 개수가 0개 혹은 2개이면 한붓그리기가 가능하다는 것은 잘 알려져 있습니다.
- ✓ 지금은 정점이 N 개인 완전그래프를 생각하고 있으므로, $N-1$ 개의 간선과 연결된 정점이 N 개 존재합니다.
- ✓ 그러므로 N 이 홀수이면 홀수 차수 정점이 없으므로 한붓그리기가 가능하며, N 이 홀수일 때의 답은 $\frac{N(N-1)}{2}$ 가 됩니다.

F. 호현이와 파이썬

- N 이 짝수이면 $N-1$ 이 홀수이므로 홀수 차수 정점이 N 개 존재하게 되고, 한붓그리기는 불가능합니다.
- 그 대신, 간선들을 한붓그리기가 가능한 여러 묶음으로 나눠 봅시다.
- 각각의 묶음은 한붓그리기가 가능하고, 한 묶음에서 다른 묶음으로 넘어갈 때마다 한 번씩 간선을 추가적으로 타게 됩니다.
- 그렇다면 필요한 묶음의 최소 개수는 얼마일까요? 한 묶음을 처리하면 항상 홀수 차수 정점 2 개를 짝수 차수 정점으로 만들 수 있으니 $\frac{N}{2}$ 개의 묶음이 필요합니다.
- 그러므로 N 이 짝수인 경우 추가적으로 간선을 $\frac{N}{2}-1$ 번 더 타야 하고, N 이 짝수일 때의 답은 $\frac{N(N-1)}{2}+\frac{N}{2}-1$ 입니다.

G. 멀티팩토리얼

`dp`, `mathematics`, `square root decomposition`

출제진 의도 – **Hard**

- 제출 30번, 정답 23명 (정답률 80.000%)
- 처음 푼 사람: **ncy09**, 9분
- 출제자: **오윤석(oh040411)**

G. 멀티팩토리얼

- 서브태스크 1
- 주어진 식을 $f(k, n) = n(n-k)(n-2k)\cdots$ 으로 정의합시다. $f(k, n)$은 아래와 같은 점화식을 만족합니다.
- $$f(k,n) = \begin{cases} n & (n \le k) \\ n \times f(k, n-k) & \text{otherwise} \end{cases}$$
- $300 \times 100\,000$ 크기의 DP 테이블 $\mathrm{dp}[k][n] = f(k, n)$을 $\mathcal{O}(NK)$에 채울 수 있고, 그 후 쿼리마다 $\mathcal{O}(1)$에 답할 수 있습니다.

G. 멀티팩토리얼

- 서브태스크 2
- $f(k, n) = n(n-k)(n-2k)\cdots$ 를 나이브하게 계산하는 시간 복잡도는 $\mathcal{O}(n/k)$ 입니다. 이는 k 가 클 때 빠르게 동작함에 주목합시다.
- 쿼리의 K 가 300 이하이면 서브태스크 1의 DP 테이블을 사용하고, 300 보다 크면 나이브하게 계산해 봅시다.
- 시간 복잡도는 모든 쿼리의 K 가 300 이하라면 $\mathcal{O}(300N + Q)$, 모두 300 보다 크면 $\mathcal{O}(QN/300)$ 입니다.
- 따라서 총 시간 복잡도는 $\mathcal{O}((300 + Q/300)N + Q)$ 로, $N = Q = 100\,000$ 인 제한에서 충분히 빠르게 동작합니다.

G. 멀티팩토리얼

- 왜 하필 300이 기준일까요?
- $B + Q/B$의 값은 산술-기하 평균 부등식에 의해 $B \approx \sqrt{Q}$ 근처에서 최소이므로, B를 $\sqrt{100\,000} \approx 316$ 근처의 아무 값이나 잡아도 됩니다. 이 경우 시간 복잡도는 $\mathcal{O}(N\sqrt{Q} + Q)$가 됩니다.
- 이와 같이 어떤 제곱근 값을 기준으로 다른 풀이를 사용하는 테크닉을 **제곱근 분할법**이라고 합니다.

H. 매드 맥스

`greedy`, `sorting`, `two pointer`, `ad-hoc`

출제진 의도 – **Hard**

- 제출 43번, 정답 19명 (정답률 44.186%)
- 처음 푼 사람: **ncy09**, 18분
- 출제자: **오윤석(oh040411)**

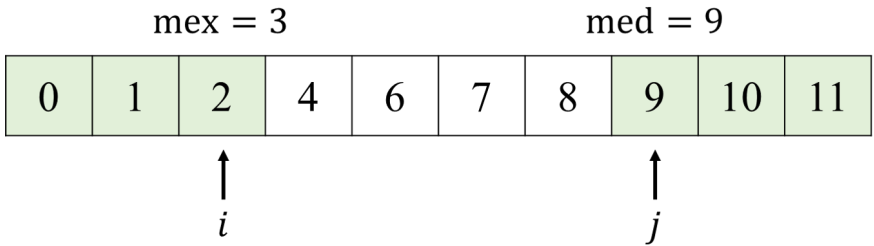
H. 매드 맥스

- Codeforces 등지에서 자주 보이는 토픽을 소재로 한 문제입니다. 크게 두 가지 관점에서 접근해볼 수 있습니다.
  1. mex를 $0, 1, 2, \cdots$로 고정하고 med를 최대화
  2. med를 $1, 2, 3, \cdots$번째로 큰 수로 고정하고 mex를 최대화
- 둘은 결론적으로 같은 풀이이지만, 증명이 상대적으로 간단한 2번 풀이를 기준으로 설명하겠습니다.
- 어떤 부분수열 B를 골랐을 때, $\mathrm{mex}(B)$와 $\mathrm{med}(B)$의 값은 A의 순서와 관계없이 변하지 않습니다.
- 분석의 편의를 위해 A를 오름차순으로 정렬하고 시작할 수 있습니다.

H. 매드 맥스

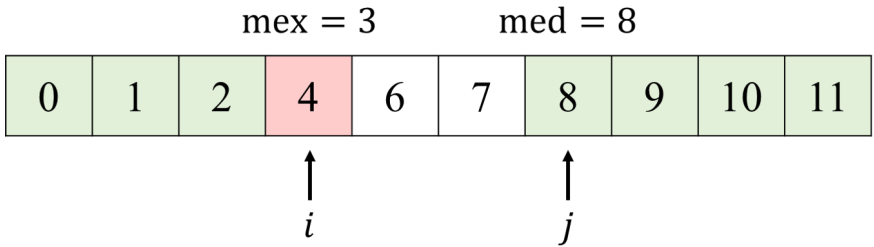
- ✓ $\mathrm{med}(B)$를 $A[j]$ (0-based)로 고정해 봅시다.
- ✓ 이때 $\mathrm{mex}(B)$를 최대화하려면 B에 $0, 1, 2, \cdots$ 가 포함되어야 합니다. 이는 A에서 가장 작은 원소 n개를 포함시켜야 달성 가능합니다.
- ✓ B에 $A[j]$보다 작은 원소 n개가 포함되면, $\mathrm{med}(B)$를 유지하기 위해 $A[j]$ 이상의 원소 n개 또한 포함되어야 합니다.
- ✓ $n = N - j$일 때, B에는 가장 작은 원소 n개와 가장 큰 원소 n개가 포함됩니다. 이때 $\mathrm{mex}(B)$는 최대가 됩니다.
- ✓ $B = A[0..i] \cup A[j..N-1]$이라고 하면, $i + j = N - 1$을 만족합니다.

H. 매드 맥스



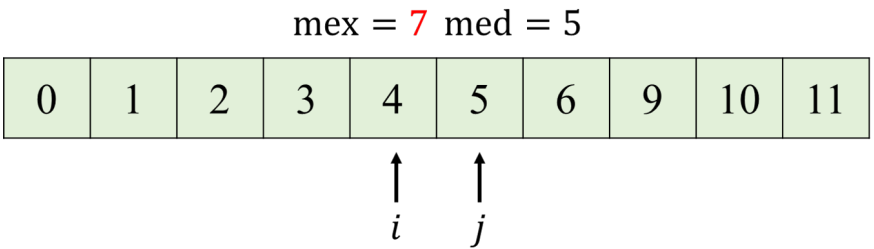
- ✓ $(i, j) = (0, N-1)$ 에서 시작하여 i 를 1 씩 증가시키고 j 를 1 씩 감소시키면서 $\text{mex}(B)$ 가 계속 증가하는지 확인할 수 있습니다.
- ✓ 현재 $A[i] = i$ 라면 $\text{mex}(B)$ 가 i 에서 $i+1$ 로 증가한 것입니다.

H. 매드 맥스



- ✓ 현재 $A[i] > i$ 이면, $\mathrm{mex}(B)$ 는 최댓값을 달성한 것이며, 더 이상 증가할 수 없습니다.
- ✓ $\mathrm{med}(B)$ 가 현재보다 큰 경우는 이미 탐색을 완료했고, 현재보다 작은 경우는 정답이 감소하는 경우뿐이므로 탐색을 중단해도 됩니다.

H. 매드 맥스



- ✓ 탐색이 중단되지 않았다면, 이러한 탐색을 $i \geq j - 1$ 일 때까지 진행합니다.
- ✓ $i \geq j - 1$ 이 되면 양쪽 구간이 붙어서 $\mathrm{mex}(B)$ 가 $\mathrm{mex}(A)$ 까지 증가할 수 있습니다.
- ✓ 역시 $\mathrm{mex}(B)$ 는 더 이상 증가할 수 없으므로 j 가 더 작은 경우는 보지 않아도 됩니다.

H. 매드 맥스

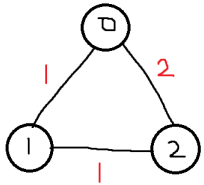
- ✓ 지금까지 본 모든 경우에 대해서 $\mathrm{med}(B) + \mathrm{mex}(B)$ 의 최댓값이 정답이 됩니다.
- ✓ 시간 복잡도는 정렬에 필요한 $\mathcal{O}(N \log N)$ 입니다.
- ✓ 한편, 잘 생각해 보면 탐색 과정에서 med 는 항상 1 이상 감소하지만 mex 는 항상 1 이하로 증가합니다. 유일한 예외는 $B = A$ 가 될 때입니다.
- ✓ 따라서 중간 과정은 사실은 고려할 필요가 없고, $B = [\min A, \max A]$ 일 때와 $B = A$ 일 때 중에서 $\mathrm{med} + \mathrm{mex}$ 가 큰 쪽이 답이 됩니다.

I. 아름다운 수열 만들기

`ad-hoc, constructive`
출제진 의도 – **Challenging**

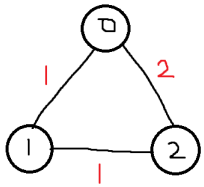
- ✓ 제출 27번, 정답 20명 (정답률 77.778%)
- ✓ 처음 푼 사람: **kky085437**, 10분
- ✓ 출제자: **김형진(plast)**

I. 아름다운 수열 만들기



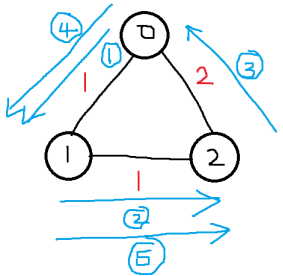
- ✓ $A = (1, 2, 0, 2, 1, 0)$ 이라는 배열이 있을 때, 위 삼각형 그림을 생각해 봅시다.
- ✓ 배열에서 $1 - 2\,(2 - 1)$ 는 각각 1 과 2 에 1 씩을 기여합니다. 따라서 0 에는 상대적으로 -1 을 기여하는 것과 같습니다.
- ✓ 비슷하게, $0 - 1\,(1 - 0)$ 은 2 에 상대적으로 -1 을 기여하고, $0 - 2\,(2 - 0)$ 는 1 에 상대적으로 -2 를 기여합니다.

I. 아름다운 수열 만들기



- 여기서 하나 알 수 있는 점은, 모든 숫자가 동일한 총합을 가지기 위해서는 상대적인 기여가 모두 같아야 한다는 점입니다.
- 따라서, $-1, -1, -2$의 기여가 각각 $2:2:1$의 비율을 가져야 한다는 점을 알 수 있습니다. 따라서, 수열은 $5k+1$의 길이를 가져야만 합니다.
- 그리고, 수열의 길이가 $5k+1$이면 항상 아름다운 수열을 만들 수 있음을 보일 수 있습니다.

I. 아름다운 수열 만들기



- ✓ 그림처럼 $0-1-2-0-1-2$라는 순서대로 이동하면, 0에서 시작해서 5개의 원소를 추가해 아름다운 수열을 만들고 2로 이동하게 됩니다.
- ✓ 해당 과정은 거꾸로도 진행될 수 있습니다. $2-1-0-2-1-0$를 이용하여, 2에서 시작해서 5개의 원소를 추가해 0으로 이동하게 됩니다.
- ✓ 따라서, 원하는 수열의 길이에 도달할 때까지 현재가 0이라면 $0-1-2-0-1-2$를, 2라면 $2-1-0-2-1-0$을 이어붙이는 방법으로 길이 $5k+1$인 수열을 모든 $k>0$에 대해 만들 수 있습니다.

J. Bracket Sequence Endgame

`dp`, `combinatorics`

출제진 의도 – **Challenging**

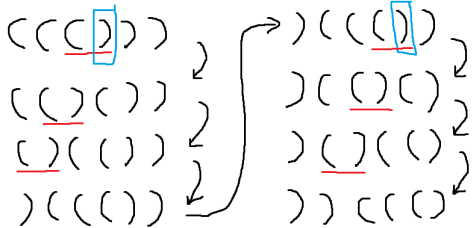
- 제출 17번, 정답 12명 (정답률 70.588%)
- 처음 푼 사람: **sadtreap**, 24분
- 출제자: **김태윤(ystaeyoon113)**

J. Bracket Sequence Endgame

- RBS에 문제에서 주어진 연산을 적용하고 나면 어떤 변화가 있는지를 먼저 관찰해 봅시다.
- (( ) ( )) 에 연산을 적용해 봅시다. 먼저, RBS의 괄호들을 짝지어 보겠습니다.
- (이) 로 바뀌고) 이 (로 바뀐다는 것은 사실 짝지어진 두 괄호의 위치를 바꾼다고 생각할 수 있습니다.
- 따라서 RBS에 연산을 적용하고 나면, 해당 연산이 적용된 구간의 모든) 의 위치가 앞으로 조금씩 이동하게 됩니다.
- 이를 통해, 문자열에 주어진 연산을 적용한 결과 문자열은 원래 문자열에 비해 일부) 들이 앞으로 이동한다는 사실을 관찰할 수 있습니다.

- ✓ 그렇다면 원본 문자열에 비해 일부) 들이 앞으로 조금씩 이동한 문자열은 우리가 무조건 만들 수 있을까요?
- ✓ 그렇습니다. 다음과 같은 쉬운 방법을 통해 무조건 만들 수 있음을 증명할 수 있습니다.
- ✓ 연산을 ( ) 에 대해서만 적용할 생각입니다. 이때, 현재 문자열과 목표 문자열을 비교해서 위치가 다른 첫) 를 찾습니다.
- ✓ 해당) 가 목표 문자열의 위치와 같을 때까지 ( ) 에 대해 연산을 진행합니다. 이 과정에서) 는 1 칸 앞으로 이동하게 됩니다.

- 예를 들어, 원본 문자열이 ((()))이고 목표 문자열은))(((라고 합시다.



- 위와 같은 과정을 통해 목표 문자열을 얻어낼 수 있습니다.

- ✓ 따라서, 우리는 원본 문자열로부터 일부) 의 위치가 앞으로 이동한 문자열의 개수를 세면 됩니다.
- ✓ $dp(i, j)$: 원본 문자열의 $[1, i]$ 까지 고려했을 때, 현재) 를 j 개 배치한 경우의 수로 정의합니다.
- ✓ $i + 1$ 번째 문자로 (를 사용할지) 를 사용할지 2가지의 transition이 있습니다.
- ✓ 이때, 현재까지 만든 문자열을 원본 문자열과 비교했을 때) 를 더 적게 사용하는 진행은 불가능합니다.
- ✓ $O(N^2)$ 으로 해결할 수 있습니다.

K. 초콜릿 먹기

`math`, `dijkstra`
출제진 의도 – **Plast**

- ✓ 제출 27번, 정답 4명 (정답률 14.815%)
- ✓ 처음 푼 사람: **leinad2**, 66분
- ✓ 출제자: **김형진(plast)**

K. 초콜릿 먹기

- ✓ 시작 지점 s에서 도착 지점 t까지 가기 위해서 우리가 알아야 할 정보는 어떤 것들이 있을까요?
- ✓ 현재 위치 (y, x)와 바라보고 있는 방향 d, 현재까지 곱해진 B 값이 중요합니다.
- ✓ 이 정보들을 효과적으로 관리할 수 있다면, 다익스트라 알고리즘을 통해 t까지 가는 최단 경로를 구하면 됩니다.
- ✓ 그러나 이 정보들을 효과적으로 관리하기는 매우매우 어렵습니다.

K. 초콜릿 먹기

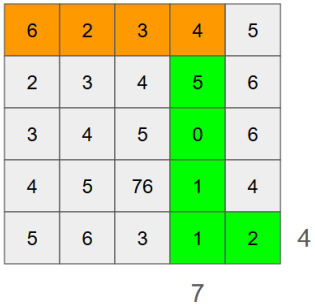
- ✓ 현재 그림에서 각 $B_{j,i}$ 는 모두 2 로 같다고 생각합시다.
- ✓ 현재 경로의 값을 계산하는 방법은 다음과 같습니다.
- ✓ $(6+2+3+4)+2(5+0+1+1)+4(2)=37$

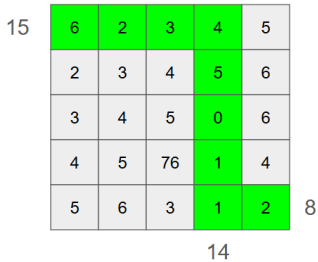
6	2	3	4	5
2	3	4	5	6
3	4	5	0	6
4	5	76	1	4
5	6	3	1	2

- ✓ s 에서 t 로 가는 과정을 거꾸로 생각해 봅시다. 즉, t 에서 s 로 간다고 생각하고 경로의 값을 계산해 봅시다.

K. 초콜릿 먹기







- ✓ t에서 s로 간다고 생각하는 경우에는 방향을 바꾸는 경우에, 지금까지 걸어온 경로의 합에 $B_{j,i}$를 곱해준다고 생각할 수 있습니다.
- ✓ 즉, $2(2(2)+5+0+1+1)+6+2+3+4=37$로 계산할 수 있습니다.

K. 초콜릿 먹기

- ✓ 따라서, t에서 s로 가는 것으로 생각할 때 우리가 알아야 할 정보는 현재 위치 (y, x)와 바라보고 있는 방향 d만 중요하게 됩니다.
- ✓ 방향을 바꿀 때는 지금까지의 최단거리에 $B_{j,i}$를 곱하게 됩니다.
- ✓ 이렇게 모델링했다면 해당 상황을 다익스트라 알고리즘을 통해서 t에서 s로 가는 최단경로를 구하면 됩니다.
- ✓ 중간중간 곱셈이 `long long` 범위를 초과할 수 있으므로 `overflow`에 조심합시다. 현재 문제의 제한 상 정답은 항상 10^{18}을 초과하지 않으므로 해당 값과 비교하여 초과하는 경우는 무시하면 됩니다.
- ✓ 시간복잡도는 $O(V \log V)$ 정도로, V는 현재 위치로 가능한 25만 개와 각각의 위치마다 방향의 개수 4개를 곱한 100만입니다.

L. 초콜릿 놓기

`constructive`
출제진 의도 – **Plast**

- ✓ 제출 8번, 정답 2명 (정답률 25.000%)
- ✓ 처음 푼 사람: **leinad2**, 127분
- ✓ 출제자: **김형진(plast)**

L. 초콜릿 놓기

- K번의 테스트케이스를 만드는 문제입니다.

L. 초콜릿 놓기

- ✓ 관찰 1) a수열이 0인 칸은 (비교적) 자유롭게 이동이 가능하고, 0이 아닌 칸은 (비교적) 자유롭게 이동하기 어렵습니다.
- ✓ 따라서, a가 0인 칸과 1인 칸으로 나누면, 미로 형태의 테스트케이스를 구성할 수 있습니다.

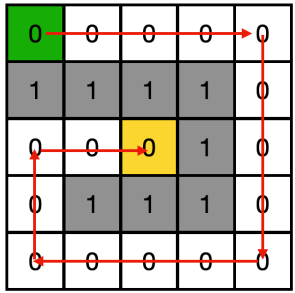
L. 초콜릿 놓기

✓ 관찰 1을 바탕으로 만들 수 있는 한 가지 방식입니다. (예제와 같습니다)

0	0	0	0	0
1	1	1	1	0
0	0	0	1	0
0	1	1	1	0
0	0	0	0	0

L. 초콜릿 놓기

✓ 최단경로의 길이는 (N+1)*(N+1)/2 정도에 수렴하며, 더 크게 만들기 어렵습니다.



- ✓ 관찰 2) b수열이 1인 칸은 (비교적) 자유롭게 회전이 가능하고, 1이 아닌 칸은 (비교적) 자유롭게 회전이 어렵습니다.
- ✓ 따라서, 모든 칸의 a를 0으로 하고, 도착점만 1로 합니다. 그 후 b수열을 회전이 가능한 칸만 1로 두면, 조금 더 강한 테스트케이스를 구성할 수 있습니다.

L. 초콜릿 놓기

✓ 관찰 2를 바탕으로 만들 수 있는 한 가지 방식입니다.

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

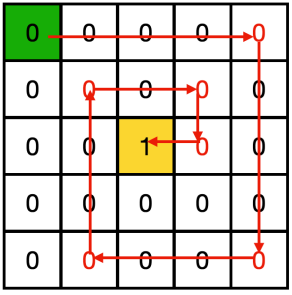
A

2	2	2	2	1
2	1	2	1	2
2	2	2	1	2
2	2	2	2	2
2	1	2	2	1

B

L. 초콜릿 놓기

✓ 최단경로의 길이는 N*N정도에 수렴하며, N*N보다 크게 만드는 것은 어렵습니다.



A

2	2	2	2	1
2	1	2	1	2
2	2	2	1	2
2	2	2	2	2
2	1	2	2	1

B

- ✓ 관찰 3) 관찰 1과 관찰 2를 동시에 적용합니다.
- ✓ 따라서, 관찰 2)와 동일한 방식을 먼저 취해봅시다.
- ✓ 모든 칸의 a를 0으로 하고, 도착점만 1로 합니다. 그 후 b수열을 회전이 가능한 칸만 1로 둡니다.
- ✓ 한 술 더 떠서, 절대로 가면 안되는 칸을 지정해, a의 값을 0보다 크게 설정하면, 기존보다 더 강한 테스트케이스를 구성할 수 있습니다.

L. 초콜릿 놓기

✓ 관찰 3을 바탕으로 만들 수 있는 한 가지 방식입니다.

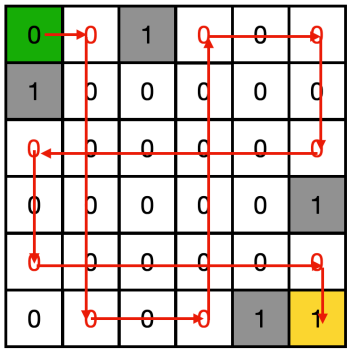
0	0	1	0	0	0
1	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	1
0	0	0	0	0	0
0	0	0	0	1	1

A

2	1	2	1	2	1
2	2	2	2	2	2
1	2	2	2	2	1
2	2	2	2	2	2
1	2	2	2	2	1
2	1	2	1	2	2

B

- 참고로 해당 예시는 N*N을 초과하지는 않습니다. 하지만 약간의 변주를 주면 N*N을 초과할 수 있게 됩니다. (직접 풀어보세요!)



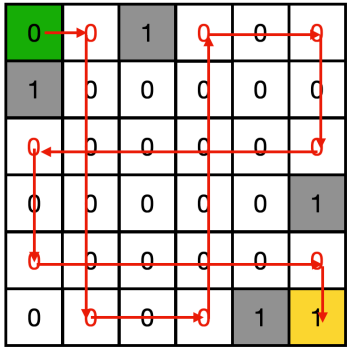
A

2	1	2	1	2	1
2	2	2	2	2	2
1	2	2	2	2	1
2	2	2	2	2	2
1	2	2	2	2	1
2	1	2	1	2	2

B

- 초콜릿 농기

✓ 해당 방식에서 응용을 잘 가할 경우, 최단경로의 길이는 N*N*2정도에 수렴하며, 이 문제를 풀 수 있는 적절한 테스트케이스가 됩니다.



A

2	1	2	1	2	1
2	2	2	2	2	2
1	2	2	2	2	1
2	2	2	2	2	2
1	2	2	2	2	1
2	1	2	1	2	2

B