2022 제 1 회 미적확통컵 에디토리얼

ai4youej, andyjung2104, azberjibiou, heeda0528, jh05013, jjang36524, kiwiyou,
wider93, gs20036, parkky, eaststar, hibye1217, bnb2011, ekwoo, qwerasdfzxcl

문제 목록

분야	번호	문제 이름	예상난이도	출제자
미적분	A	연속인가? ?	Bronze	haru_101
	B	수열의 극한값	Silver	jjang37524
	C	함수와 MST	Platinum	ai4youej
	D	다항함수의 적분과 쿼리	Platinum	Aiwiyou
	E	연립방정식	Platinum	andyjung2104
	F	이차함수와 직선	Diamond	azberjibiou
확률과 통계	G	균등분포와 정규분포	Silver	jh05013
	H	방향 정하기	Gold	andyjung2104
	I	빙고	Gold	wider93
	J	살얼음판 걷기	Platinum	heeda0528
	K	당근과 채찍	Platinum	andyjung2104
	L	이항분포에서 가장 큰 직사각형	Platinum	jh05013

후원 및 도움

YoungJin.com Y. 영진닷컴

STARTLINK

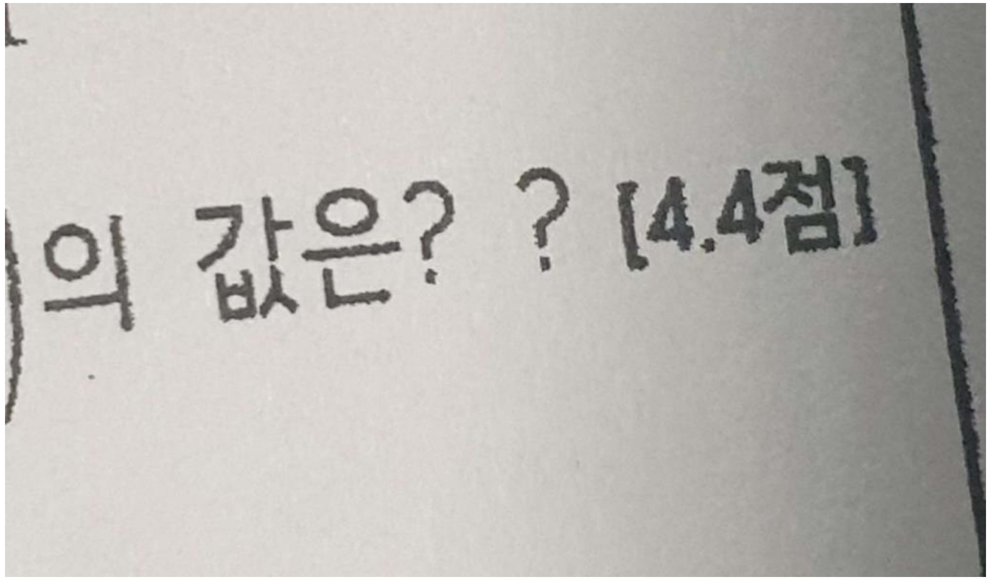
AC solved.ac

+ 그 외에 특별상 후원에 도움 주신 분들

A – 연속인가? ? haru_101

1 초 1024 MB

- 두 함수가 연속인지 확인하기 위해선, x에 k를 대입해서 두 함수의 함숫값이 동일한지 확인하면 됩니다.
- 함숫값의 범위는 $-2^{31}-1 \sim 2^{31}-1$을 초과할 수 있으므로 C++의 long long과 같은 자료형을 사용해야 합니다.
- 여담으로, 물음표가 2개가 붙은 이유는 출제자 미적분 중간고사 시험지에 있던 문제의 오타에서 영감을 받았습니다.



B – 수열의 극한값 `jjang36524`

1 초 1024 MB

- 답은 $x^2 - bx - c$의 양수근이며, 식으로는 $a[i+2] = ba[i+1] + ca[i]$입니다.

- $x[i] = \frac{a[i+2]}{a[i+1]}, y[i] = \frac{a[i+1]}{a[i]}$로 정의하면, $x[i] = b + \frac{c}{y[i]}$입니다.

- $b > 0$이면 $y[i+2] = \frac{a[i+3]}{a[i+2]} = b + \frac{cy[i]}{by[i]+c}$, 이는 $y[i]$가 $x^2 - bx - c$의 양수근보다 작을 때,

  $y[i+2] > y[i]$임을 증명할 수 있습니다.

- 따라서 $\frac{a[i+1]}{a[i]}$는 $x^2 - bx - c$의 양수근으로 수렴합니다.

C – 간단한 MST 문제 ai4youej

3 초 1024 MB

- 모든 간선의 가중치는 $at^2 + bt + c$ 꼴로 표현됩니다.

- 먼저 아래와 같은 사실들을 관찰할 수 있습니다.
  1. 모든 간선에서 a값이 동일합니다.
  2. 당연히 최소 스패닝 트리의 간선 수는 $(V - 1)$개 입니다.

- $f(t)$를 어떻게 하면 정확하게 구할 수 있을까요?

C – 간단한 MST 문제 `ai4youej`

3 초　　1024 MB

- 두 간선이 있고, 가중치가 각각 $at^2 + b_1t + c_1$, $at^2 + b_2t + c_2$이라고 합시다.
  두 간선의 가중치 대소 관계는 $at^2 + b_1t + c_1 = at^2 + b_2t + c_2$를 만족하는 t값을 기준으로 바뀝니다.
  (물론 바뀌지 않을 수도 있습니다.)

- 모든 $\frac{E(E-1)}{2}$개의 간선 쌍에 대해서 대해서 대소 관계가 바뀌는 t값을 찾아줍니다.
  위에서 구한 t값을 오름차순으로 정렬하고, 이를 $t_1, t_2, \cdots, t_k$로 이름을 붙이겠습니다.
  $t_0 = -\infty, t_{k+1} = \infty$ 이라고 정의합니다.

- 그렇다면, $i = 0, 1, \cdots, k$에 대해서 $[t_i, t_{i+1}]$에서 가중치의 대소 관계는 변하지 않습니다.
  즉, MST를 이루는 간선들은 변하지 않습니다. (MST가 여러 개여도 성립합니다.)
  $[t_i, t_{i+1}]$에서 $f(t)$=($[t_i, t_{i+1}]$에서 MST를 이루는 간선들의 가중치 합) 이 됩니다.

C – 간단한 MST 문제 `ai4youej`

3 초 1024 MB

- 이제, 우리는 $f(t)$를 쪼개서 구할 수 있습니다.

    - 직접 MST를 구하면서 간선 정보를 저장합니다.
    - 저장한 간선 정보를 전부 합쳐서 직접 $f(t)$를 구합니다.
    - $f(t)$를 $[t_i, t_{i+1}]$에 대해 정적분하고, 그 값들을 합쳐서 풀 수 있습니다.

- 근데, 간선 정보의 저장 없이 조금 더 쉽게 구할 수 있는 방법이 있습니다.

C – 간단한 MST 문제 `ai4youej`

3 초 1024 MB

- 모든 간선의 a값이 동일하므로, 이차항의 계수는 그냥 $a(V-1)$입니다.

  a 가 아닌 $a(V-1)$이라는 것에 주의해야합니다.

- 남는 부분은 일차함수 부분입니다. 일차함수 $g(x)$에 대해 아래 식이 성립함을 이용하면 더 쉽게 구현할 수 있습니다.

$$\int_a^b g(x)dx = \frac{1}{2}(b-a)(g(b)+g(a))$$

- 즉, $f(t_i)$와 $f(t_{i+1})$의 값을 구하고 위 식에 대입하면 적분값을 얻을 수 있습니다.

  $f(t_i)$와 $f(t_{i+1})$는 직접 대입해서 가중치가 상수인 그래프를 새로 만들어서 MST의 가중치의 합을 구해서 풀면 됩니다. 프림 알고리즘이나 크루스칼 알고리즘을 돌리면 됩니다.

C – 간단한 MST 문제 `ai4youej`

3 초 1024 MB

- 이제 최종 답을 구하기 위해서 쪼개서 구한 적분값을 다 더해야 합니다.
  당연히 주어진 시간 범위 내에서만 적분하면 됩니다.
  이때, $(ab^{-1}) \times (cd^{-1}) \equiv (ad + bc) \times (bd)^{-1} \ \ mod\ (10^9 + 7)$을 이용하면 됩니다.

- 전체 시간 복잡도는 다음과 같습니다.
  - 각 간선 쌍에 대해서 대소 관계가 바뀌는 t값을 찾습니다. $- O(E^2)$
  - 적분을 하기 위해서는 MST를 구하는 알고리즘을 계속 반복해서 돌려야 합니다.
    프림 알고리즘의 시간 복잡도가 $O(E \log V)$이고, 크루스칼은 $O(E \log E)$인데, 아무거나 사용해도 상관 없습니다.
    최대 $O(E^2)$번 구해야 하므로, 프림 기준 $O(E^3 \log V)$, 크루스칼 기준 $O(E^3 \log E)$의 시간 복잡도를 가집니다.

- Python의 경우 단순 fractions 모듈을 사용하면 '시간 초과'를 받을 수 있으니 유의해주세요.

D – 다항함수의 적분과 쿼리 `kiwiyou`

1 초 1024 MB

- 첫 번째와 두 번째 조건에 의해, $\forall n \in \mathbb{Z}\ (2 \le n \le N) : f_{n-1}(n-1) = f_n(n-1) = A_{n-1}$입니다.

  세 번째 조건에 의해, $\forall n \in \mathbb{Z}\ (2 \le n \le N) : f'_{n-1}(n-1) = f'_n(n-1)$입니다.

  네 번째 조건에 의해 각 f_n의 차수는 앞에서부터 차례대로 정하면 됩니다.

- f_1의 경우 A_0과 A_1이 주어져 있으므로, $f_1(x) = (A_1 - A_0)x + A_0$입니다.

  f_2부터 각 f_n은 A_{n-1}과 A_n이 주어져 있고, $f'_{n-1}(n-1) = f'_n(n-1)$이 미리 정해지므로, 2차로 충분합니다.

- 각 쿼리가 g의 정의가 나뉘는 경계에서만 적분을 요구하고 있기 때문에, 각 f_n을 구간 $[n-1, n]$에서 적분한 결과를

  저장해두고, 부분합을 구해서 풀 수 있습니다.

D – 다항함수의 적분과 쿼리 `kiwiyou`

1 초 1024 MB

- $f_n(x) = p_n\big(x - (n-1)\big)^2 + q_n\big(x - (n-1)\big) + r_n$으로 두면,

  $6 \times \int_{n-1}^{n} f_n(x)dx = 6 \times \int_0^1 p_n x^2 + q_n x + r_n\,dx = [2p_n x^3 + 3q_n x^2 + 6r_n]_0^1 = 2p_n + 3q_n + 6r_n \stackrel{\text{def}}{=} s_n$

  $f_n(n-1) = r_n = A_{n-1}$

  $f_n(n) = p_n + q_n + A_{n-1} = A_n$

  $f_n'(n-1) = q_n = f_{n-1}'(n-1) = 2p_{n-1} + q_{n-1}$

  밑의 두 식을 연립하여 $p_n + q_n = A_n - A_{n-1}$과 $q_n + q_{n-1} = 2 \times (A_{n-1} - A_{n-2})$를 얻습니다.

D – 다항함수의 적분과 쿼리 kiwiyou

1 초 1024 MB

- $S_n = q_n + A_n + 5A_{n-1}$이므로, A_n을 관리하는 세그먼트 트리, $A_n - A_{n-1}$을 홀수 / 짝수로 나누어 관리하는 세그먼트 트리를 이용하면 부분합과 값 변경 쿼리를 $O(\log n)$에 처리할 수 있습니다.

E – 연립방정식 andyjung2104

2 초　　1024 MB

$$f(x) = (x-a_1)(x-a_2)\cdots(x-a_n)$$

$a_1 < a_2 < \cdots < a_n$ $\Rightarrow \displaystyle\sum_{k=1}^{n} \frac{a_k^m}{f'(a_k)} = \begin{cases} 0 & (m < n-1) \\ 1 & (m = n-1) \end{cases}$

f는 n차다항식, $f(x)=k$의 근을 크기순으로 $\alpha_1(k), \alpha_2(k), \cdots \alpha_n(k)$라 하자.

$a_i = \alpha_i(0)$ $(1 \le i \le n)$

$f(\alpha_i(k)) = k \longrightarrow$ (k로 미분) $f'(\alpha_i(k)) \cdot \frac{d\alpha_i(k)}{dk} = 1$ $\therefore \frac{d}{dk}\alpha_i(k) = \frac{1}{f'(\alpha_i(k))}$

(0에서는 미분가능)

k에 0 대입 $\Rightarrow$ $\left.\frac{d}{dk}(\alpha_i(k))\right|_{k=0} = \frac{1}{f'(a_i)}$

양변에 $\times a_i^m$ 하면 $\alpha_i(0)^m \cdot \left.\frac{d}{dk}(\alpha_i(k))\right|_{k=0} = \frac{a_i^m}{f'(a_i)}$

$$\frac{1}{m+1}\left.\frac{d}{dk}\left(\alpha_i(k)^{m+1}\right)\right|_{k=0} = \frac{a_i^m}{f'(a_i)}$$

$\Downarrow$

$$\boxed{\frac{1}{m+1}\cdot\left.\frac{d}{dk}\left(\sum_{i=1}^{n}\alpha_i(k)^{m+1}\right)\right|_{k=0} = \sum_{i=1}^{n}\frac{a_i^m}{f'(a_i)}}$$

* $(x-a_1)(x-a_2)\cdots(x-a_n) = k$의 근들.

$\Rightarrow$ $0 \le j \le n-1$에 대해, $\underbrace{(\text{서로다른 } j\text{개 수의 곱})}_{e_j}$의 합은 k값과 무관.

$j=n$이면, $e_n = a_1 a_2 \cdots a_n + (-1)^{n-1}k$

* Newton's Identities

$$p_k(x_1, \cdots x_n) = \sum_{i=1}^{n} x_i^k$$

서.다.

$e_k(x_1, \cdots x_n) = (x_1 \sim x_n$ 중 k개 수의 곱$)$의 합

$$\Rightarrow \underline{k e_k = \sum_{i=1}^{k}(-1)^{i-1} e_{k-i} p_i}$$

$\Rightarrow p_k$는 $e_0 \sim e_k$에만 의존.

$e_n = a_1 a_2 \cdots a_n + (-1)^{n-1}k$

$n e_n = (\quad) + p_n e_0 (-1)^{-n-1}$

, $\underline{\frac{dp_n}{dk} = n.}$

$\therefore$ $i < n$이면 $\frac{dp_i}{dk} = 0$임은 자명하므로 증명 끝.

E – 연립방정식 andyjung2104

2 초 1024 MB

- 증명은 https://en.wikipedia.org/wiki/Newton%27s_identities 를 참고하면 된다.

F – 이차함수와 직선 `azberjibiou`

python/pypy : 0.5 초, Java : 0.4 초, 그 외 : 0.1 초 1024 MB

Phase 1. Azber가 이기는지에 대한 판별

- 모든 이차함수의 이차항의 계수가 동일하면 자명하게 모든 이차함수와 안 만나는 직선이 존재한다.

- 이차항의 계수가 1인 이차함수들의 집합을 S_+, 이차항의 계수가 −1인 이차함수들의 집합을 S_-라고 하자.

- 이차함수 $y = f(x)$에 접하는 기울기 m의 직선을 $y = mx + F(f, m)$으로 $F(f, m)$을 정의하자.

- $f(x) = x^2 + bx + c$에 대해 $F(f, m) = -\frac{(b-m)^2}{4} + c$이고, $f(x) = -x^2 + bx + c$에 대해 $F(f, m) = \frac{(b-m)^2}{4} + c$이다.

F – 이차함수와 직선 `azberjibiou`

python/pypy : 0.5 초, Java : 0.4 초, 그 외 : 0.1 초 1024 MB

- S_+에 속하는 모든 $f(x) = x^2 + bx + c$에 대해서
  $g_+(m) = \min\left(F(f, m) + \frac{1}{4}m^2\right) = \min\left(\frac{1}{2}bm + c - \frac{1}{4}b^2\right)$으로 정의하자.

- S_-에 속하는 모든 $f(x) = -x^2 + bx + c$에 대해서
  $g_-(m) = \max\left(F(f, m) - \frac{1}{4}m^2\right) = \max\left(-\frac{1}{2}bm + c + \frac{1}{4}b^2\right)$으로 정의하자.

- 어떤 m, k가 존재해서 $\frac{1}{4}m^2 + g_-(m) < k < -\frac{1}{4}m^2 + g_+(m)$를 만족하면 $y = mx + k$는 모든 이차함수와 만나지 않는다.

- 반대로 모든 m에 대해서 $h(m) = g_+(m) - g_-(m) \leq \frac{1}{2}m^2$를 만족하면 임의의 직선은 하나 이상의 이차함수와 만나게 된다.

F – 이차함수와 직선 azberjibiou

python/pypy : 0.5 초, Java : 0.4 초, 그 외 : 0.1 초 1024 MB

- $h(m)$은 위로 볼록한 함수이기 때문에, $h(m)$과 $\frac{1}{2}m^2$을 비교하기 위해서는 $h(m)$을 이루는 선분 / 반직선 / 직선들과 $\frac{1}{2}m^2$을 비교할 수 있으면 된다.

- $h(m)$을 이루는 직선은 최대 N개이기 때문에 $h(m)$의 개형을 $O(NlogN)$에 만든 뒤에 $h(m)$과 $\frac{1}{2}m^2$과의 비교를 $O(N)$에 마칠 수 있다.

F – 이차함수와 직선 `azberjibiou`

python/pypy : 0.5 초, Java : 0.4 초, 그 외 : 0.1 초 1024 MB

Phase 2. Azber가 이긴다면 사용하는 이차함수의 개수는 3개 이하이다.

- Azber가 이기는 경우에 대해서만 생각해주자. 이차함수를 1개 사용하는 경우는 자명하게 없다.
  이차항이 1인 이차함수와 이차항이 −1인 이차함수 두 개가 만나는 경우에는 2개의 이차함수를 그려서 Azber가 이길 수 있다.

- $y = x^2 + b_1x + c_1$와 $y = -x^2 + b_2x + c_2$가 만나기 위해서는 $(b_1 - b_2)^2 - 8(c_1 - c_2) \geq 0$이어야 한다.
  이는 식 변형을 통해 $\frac{1}{8}(b_1^2 - 2b_2b_1 + b_2^2) + c_2 \geq c_1$이 된다.

- 이차항의 계수가 −1인 모든 이차함수에 대해 $-\frac{1}{4}b_2x + \frac{1}{8}b_2^2 + c_2$에 대한 convex hull을 만든 뒤에 이차항의 계수가 1인 모든 이차함수를 b_1을 기준으로 정렬하고 스위핑을 하면 이차항의 계수가 다른 두 이차함수가 만나는지를 $O(NlogN)$에 판별할 수 있다.

F – 이차함수와 직선 azberjibiou

python/pypy : 0.5 초, Java : 0.4 초, 그 외 : 0.1 초 1024 MB

- Azber가 3개 이상의 이차함수를 사용하는 경우에 대해서 생각을 하자.

- $h(m)$을 이루는 선분 / 반직선 / 직선 중에 $y = \frac{1}{2}x^2$과 만나지 않는 직선 $y = l(m)$이 있다고 가정하자.

  $f_1 \in S_+,\ f_2 \in S_-$인 f_1, f_2가 존재해서 $l = F(f_1, m) + \frac{1}{4}m^2 - (F(f_2, m) - \frac{1}{4}m^2)$으로 표현될 수 있다.

- f_1, f_2만 좌표평면에 그린다면 좌표평면에 그린 이차함수에 대해서 구한 $h(m)$은 $y = l(m)$의 형태로 나타난다.

  이는 2개의 이차함수만 좌표평면에 그려서 임의의 직선이 적어도 하나의 이차함수와 만나게 했다는 것으로,

  가정에 모순이다.

- 따라서 $h(m)$을 이루는 직선은 반드시 $y = \frac{1}{2}m^2$과 만난다.

F – 이차함수와 직선 azberjibiou

python/pypy : 0.5 초, Java : 0.4 초, 그 외 : 0.1 초 1024 MB

- $h(m)$ 상에 있는 인접한 두 선분이 존재해서 왼쪽 선분은 오른쪽으로 연장했을 때 $y = \frac{1}{2}m^2$과 만나고,

  오른쪽 선분은 왼쪽으로 연장했을 때 $y = \frac{1}{2}m^2$과 만난다고 가정하자.

- $h(m)$의 두 선분을 이루는 이차함수는 3개로, 이들을 좌표평면에 그리면 임의의 직선은 한 이차함수와 만나게 할 수 있다. 따라서 답은 3 이하이다.

- 실례는 $h(m)$을 이루는 직선들을 따라가면서 구할 수 있고, 이는 $h(m)$이 이미 만들어졌다는 가정 하에 $O(N)$에 구할 수 있다. 따라서 정답을 $O(NlogN)$에 구할 수 있다.

F – 이차함수와 직선 azberjibiou

python/pypy : 0.5 초, Java : 0.4 초, 그 외 : 0.1 초 1024 MB

- 구현이 까다롭고, 오버플로우를 조심해서 구현해야 한다. long long 범위 내에서 문제를 해결할 수 있지만, C++에서는 __int128을 사용하는 것도 좋은 방법이다.

- TL은 굉장히 느슨하다. (정해는 8ms) 시간복잡도만 $O(NlogN)$이면 충분히 AC를 받을 수 있을 것이다.

G - 균등분포와 정규분포 jh05013

1 초 1024 MB

- 육안으로 구별할 수 없는 동전 2개가 있는데, 하나는 앞면이 나올 확률이 10%고 하나는 50%라고 합시다. 둘 중 하나를 골랐을 때, 이 동전이 10%짜리 동전인지 아닌지 알아내려면 어떻게 할까요? 직접 던져보면 됩니다.
- 1,000번쯤 던졌을 때 앞면이 100회 근처로 나왔으면 이 동전은 1에 가까운 확률로 10%짜리, 아니면 50%짜리 동전입니다. 물론 50%짜리 동전인데 앞면이 100회만 나올 수도 있지만, 그럴 확률은 복권 몇 개가 연이어서 당첨될 확률과 같습니다.
- 이 문제에서도 두 분포에서 확률이 뚜렷하게 차이 나는 조건을 하나 찾아, 그 조건에 해당되는 수가 몇 번 나왔는지 세면 됩니다. 예를 들어 출제자의 풀이는 0.05 이하의 값이 180개 이상이면 A, 아니면 B라고 판단합니다.
- 지문에서 데이터를 실제로 두 방법 중 하나를 써서 만들었다고 했으므로, 1에 가까운 확률로 맞는 풀이면 통과할 것입니다.

G – 균등분포와 정규분포 jh05013

1 초 1024 MB

- 여기서 끝내면 재미없으니, 실제로 위 풀이가 1에 가까운 확률로 맞는다는 증명을 첨부합니다.
- Suppose the answer is A (uniform distribution). The algorithm guesses B iff $X < 180$ where X follows a binomial distribution $B(5000, 0.05)$.
- Since n is very large, this can be approximated as $N(250, 237.5)$, so suppose $Y \sim N(250, 237.5)$.
- $P(Y < 180) = P(Z < \frac{180-250}{\sqrt{237.5}}) \approx P(Z < -4.542) \approx 0.00000279$ where Z follows the standard normal distribution. Therefore, the algorithm guesses B with probability $< p$, where $p = 0.000003$.
- Suppose the answer is B. Let $T \sim N(0.5, 0.1)$. The probability that the observed value is < 180 is $\frac{P(0 \le T \le 0.05)}{P(0 \le T \le 1)}$.

G – 균등분포와 정규분포 jh05013

1 초 1024 MB

- $P(0 \leq T \leq 1) = P\left(\frac{0-0.5}{\sqrt{0.1}} \leq Z \leq \frac{0.05-0.5}{\sqrt{0.1}}\right) \approx P(-1.58 \leq Z \leq 1.58) \approx 0.8858$

- $P(0 \leq T \leq 0.05) = P(\frac{0-0.5}{\sqrt{0.1}} \leq Z \leq \frac{0.05-0.5}{\sqrt{0.1}}) \approx P(-1.58 \leq Z \leq -1.42) \approx 0.0207$

- So $\frac{P(0 \leq T \leq 0.05)}{P(0 \leq T \leq 1)} \approx 0.0233$.

- The algorithm guesses A iff $X \geq 180$ where X follows a binomial distribution $B(5000, 0.0233)$.

  Since n is very large, this can be approximated as $N(116.5, 113.8)$, so suppose $Y \sim N(116.5, 113.8)$.

G – 균등분포와 정규분포 jh05013

1 초 1024 MB

- $P(Y \geq 180) = P\left(Z \geq \frac{180-116.5}{\sqrt{113.8}}\right) = P(Z \geq 5.952)$ where Z follows the standard normal distribution.

  Therefore, the algorithm guesses A with probability $\ll p$.

- We're given 100 testcases. Suppose $X \sim B(100, 10^{-6})$, then

  $P(X \leq 1) = P(X = 0) + P(X = 1) = (1-p)^{100} + 100p(1-p)^{99} = (1-p)^{99}(1+99p) \approx 0.999999955.$

  Therefore, the probability of wrong answer is smaller than 10^{-7}.

H – 방향 정하기 `andyjung2104`

1 초 1024 MB

- 정답은 $N!$ 입니다.
  이는 직관적으로, 점들을 일렬로 세울 때마다 한 가지 씩 방법이 나온다고 생각해서 답을 맞출 수 있습니다.
  엄밀하게 증명을 해봅시다.

- 정답이 일렬로 순서를 정하는 방법 뿐임을 보이자.
  outdegree가 가장 큰 점 중 하나를 A라 하면, B->A인 점 B가 존재했을 때 A->C일 때마다 B->C여야 해서 (아니면 ACBA 회로 존재) B의 outdegree가 A의 outdegree보다 크다. 모순.
  따라서 A는 모든 점으로 나간다.

- A를 제외한 점 중 outdegree가 가장 큰 점을 B 라 하면, 같은 논리로 B는 (A를 제외한) 모든 점으로 나간다.
  이를 계속 반복하면 결국 A->B->C->D->...로 앞선 점에서 뒤쪽 모든 점으로 간선이 나가는 그래프가 된다.
  따라서 A, B, C, D, … 처럼 점들의 순서를 정하는 것과 일대일대응된다.

I – 빙고 wider93

1 초 1024 MB

- 기댓값의 선형성에 의해, 각 줄의 빙고가 완성될 확률을 따로따로 구해서 더해주면 됩니다.

  N개의 칸 중 정확히 m개의 칸이 비어 있는 줄을 생각해 봅시다. N^2개의 칸 중 M칸이 비어 있는 상황이라고 합시다.

  앞으로 K개를 칠하게 되는데, N칸 모두가 크기 K의 부분집합에 속하는 확률을 구해주면 됩니다.

- 이 확률은 여러 방법으로 계산할 수 있지만, 출제자에게 제일 재미있는 해석은 M개의 칸에 1부터 M까지 번호를 주었을 때 우리가 고정한 줄에 있는 m개의 칸의 번호가 모두 K 이하가 되도록 하는 것입니다.

  따라서 $\frac{K(K-1)\cdots(K-m+1)}{M(M-1)\cdots(M-m+1)}$ 이 우리가 구하는 확률이 됩니다.

- 이제 각 줄에서 색이 채워지지 않은 칸의 수를 세어서 위 값에 $(N^2)!$을 곱해 준 값들을 계산하고, 다 더해주면 답을 얻을 수 있습니다.

J – 살얼음판 걷기 heeda0528

1 초 1024 MB

- 즈티가 살얼음판을 왕복하는 상황을 1번 판에서 두 명이 출발하여 각각 N번 칸에 도착하되, 중간에 같은 판을 지나지 않는 상황으로 바꿔서 생각할 수 있습니다.

- 두 사람의 이동 순서는 중요하지 않기 때문에, 겹치는 경우가 생기지 않도록 경우의 수를 '잘' 세줘야 합니다.

- 이를 위해 '가장 최근에 이동한 사람이 N번 칸과 더 가깝게 위치하고, 두 사람이 처음과 마지막을 제외하고 K칸 이상 떨어지지 않도록' 이동을 강제합시다.

- $dp(i, j)$: 가장 최근에 이동한 사람이 i번 칸에 위치하고, 두 사람이 j칸 떨어진 상황이 되기 위한 경우의 수 라고 정의합시다.

J – 살얼음판 걷기 heeda0528

1 초 1024 MB

- 정의에 의해 $dp(1,0) = 1, dp(i,0) = 0\ (1 < i), answer = dp(n,0)$ 입니다.
- 가장 최근의 이동을 고려하면, $dp(i,j) = dp(i-1,j-1) + dp(i-2,j-2) + \dots + dp(i-j+1,1) + dp(i-j,0) + dp(i-j,1) + \dots dp(i-j,k-j)$ 의 점화식을 얻을 수 있습니다.
- 식을 정리하면 $dp(i,j) = \sum_{t=1}^{j-1} dp(i-j+t,t) + \sum_{t=0}^{k-j} dp(i-j,t)$이므로, 가로 누적합과 대각선 누적합을 각각 관리하면 각 항을 $O(1)$, 전체 문제를 $O(NK)$에 해결할 수 있습니다.
- 상황을 어떻게 설정하냐에 따라 다양한 풀이가 나올 수 있습니다.

K – 당근과 채찍 andyjung2104

4 초 1024 MB

- 당근은 $+b$, 채찍은 $-a$ 만큼 기분을 변화시킨다.

  어떤 배열에 대해, 행동횟수에 따른 기분 그래프를 그려보면, 시작점과 끝점이 0인 어떤 그래프가 나올 것이다.

- 이 배열을 한 칸씩 회전해 보면, 그래프의 최저점이 시작점에 올 때만 기분이 모든 점에서 0 이상이 된다.

  따라서 배열 $(a+b)$개마다 1개씩 세면 되고, 즉 답은 $a +_b C_a \times \frac{1}{a+b}$이다.

- 이제 이를 빠르게 계산하자. (쿼리당 O(log(a+b)) 시간으로)

  식변형을 하면, 답은 $\frac{(a+b-1)!}{a!b!} = (a+b-1)! \times (a!)^{-1} \times (b!)^{-1}$ 이다.

K – 당근과 채찍 andyjung2104

4 초 1024 MB

- $a, b \le 10^6$ 이므로 $1! \sim (2 \times 10^6)!$ 을 $10^9 + 7$로 나눈 나머지를 미리 구해 놓으면 (전처리 $O(a+b)$)
  이 쿼리에서 계산해야 하는 것은 모듈러 역원 두 개 뿐이다.

- 따라서 페르마 소정리 혹은 확장 유클리드 호제법을 사용하면, $O(\log(a+b))$ 시간에 쿼리를 해결할 수 있다.
  여기까지만 하면 문제를 해결할 수 있다. 총 시간복잡도 $O(a+b+Qlog(a+b))$

- 쿼리당 상수시간에 계산해보자. 이는 단순히 팩토리얼의 모듈러 역원들을 미리 구해주면 된다.
  $1!, 2!, 3!, \ldots, (2 \times 10^6)!, (2 \times 10^6)!^{-1}, (2 \times 10^{-6} - 1)^{-1}, \ldots$ 순서대로 구해주면 전처리를 $O(a+b)$에 할 수 있다.
  ($(a+1)!^{-1} \times (a+1) = a!^{-1}$ 이기 때문에)

- 이제 각 쿼리를 $O(1)$에 풀 수 있으므로, 총 시간복잡도 $O(a+b+Q)$

L – 이항분포에서 가장 큰 직사각형 `jh05013`

1.5 초 1024 MB

- 우선 히스토그램 자체를 구해야 합니다.

  이것도 생각보다 어려운데, 첫 번째 막대부터 차례대로 $\frac{(n-i)p}{i(1-p)}$ 를 곱해서 구하면 안 됩니다.

  N이 클 경우 부동소숫점의 원리에 따라 첫 번째 막대의 높이가 정확히 0으로 계산되어서,

  나머지 막대도 전부 0이 되기 때문입니다.

- 일반적으로 0에 가까운 양수나 반대로 매우 큰 양수를 다루는 것은 위험합니다.

  다음 입력을 넣었을 때 0이 출력된다면 이 함정에 빠진 것일 가능성이 큽니다.

```
200000 0.5 0
1 200000
```

L – 이항분포에서 가장 큰 직사각형 `jh05013`

1.5 초 1024 MB

- 높이를 제대로 구하려면 곱하고 나누는 순서를 잘 조정해서 도중에 0 근처로 가지 않게 하거나,
  로그를 씌워서 더하고 뺀 다음 exp를 씌워서 되돌려야 합니다.
  로그를 사용할 경우 C++과 Python 등에서 기본으로 제공하는 lgamma 함수를 사용하면 편합니다.

- 이제 쿼리를 처리합시다.
  일반적으로 가장 큰 직사각형에 대한 구간 쿼리는 매우 어려운 문제이기 때문에,
  이 히스토그램의 특수성을 잘 활용해야 합니다.

- N이 작다면 히스토그램이 증가하다가 감소하는 형태라는 점을 활용하여 모든 구간의 답을 $O(N^2)$ DP로
  미리 계산할 수 있습니다.

L – 이항분포에서 가장 큰 직사각형 jh05013

1.5 초 1024 MB

- N이 클 때가 문제인데, 이항 확률변수가 베르누이 확률변수 n개의 합이라는 사실을 상기합시다.
  큰 수의 법칙에 의해 이 히스토그램은 굉장히 "얇습니다." 즉, x가 평균 np에서 조금만 벗어나도 x번째 막대의 높이는 0에 가깝습니다. 그런 막대를 사용하면 직사각형의 넓이도 0에 가깝기 때문에,
  평균에 가까운 위치의 막대가 존재하는 한 가장 큰 직사각형이 될 수 없습니다.

- 따라서 평균에서 어느 정도 벗어나는 막대는 아예 없다고 가정하고, 나머지를 $O(N^2)$ DP로 미리 처리하면 됩니다.
  $\pm 2{,}000$ 정도로 잡으면 충분합니다.

L – 이항분포에서 가장 큰 직사각형 `jh05013`

1.5 초 1024 MB

- 평균에 가까운 위치의 막대는 높이를 직접 구하는데, 이때 오차를 2^{-57} 이하로 줄일 수 있습니다. (https://github.com/lattera/glibc/blob/895ef79e04a953cac1493863bcae29ad85657ee1/sysdeps/ieee754/dbl-64/e_lgamma_r.c#L58)

- 또한 평균에서 2,000 이상 벗어나는 막대는 높이를 0으로 간주하는데, 이때 $np(1-p)$는 $n = 200{,}000, p = \frac{1}{2}$일 때 약 111.8로 최대가 되고,

- $P(X \geq np + 2000)$

  $= P(2000 \leq Y)$ where $Y \sim N(0, np(1-p))$

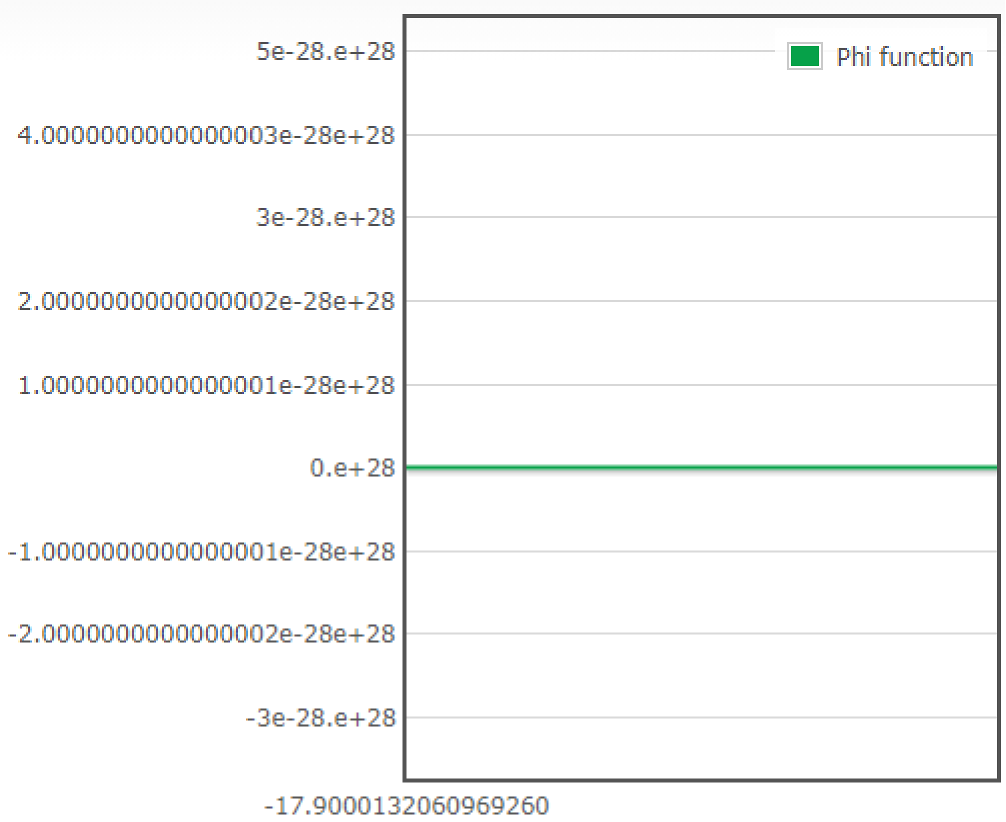
  $= P(\frac{2000}{\sqrt{np(1-p)}} \leq Z)$ where $Z \sim N(0,1)$

  $\leq P(17.88 \leq Z)$.

L – 이항분포에서 가장 큰 직사각형 jh05013

1.5 초 1024 MB

- 이 값은 너무 작아서 웬만한 웹사이트에 있는 계산기도 자세하게 계산을 안 해 주지만, 2^{-57}보다 작은 것은 확실합니다. 막대가 최대 4,000개이므로 직사각형 넓이의 오차는 10^{-13} 이하입니다.



L – 이항분포에서 가장 큰 직사각형 jh05013

1.5 초 1024 MB

- <별해> 출제자의 최초 정해는 히스토그램이 증가하다가 감소하는 형태라는 점만 사용했습니다.

- 높이가 가장 높은 지점을 m이라고 합시다. 만약 쿼리가 m을 포함하지 않을 경우, $[l, r]$ 구간은 높이가 항상 증가하거나 항상 감소하는 형태입니다. 증가하는 형태일 경우 직사각형의 오른쪽 끝은 r이고, 왼쪽 끝을 x라고 할 때 넓이는 $A[x] \times (r - x + 1)$입니다. 모든 $l \leq x \leq r$에 대해 이 식의 최댓값이 답입니다. 감소하는 형태도 비슷합니다.

- 쿼리가 m을 포함할 경우, 막대 m에서 시작해서 좌우로 한 칸씩 뻗되, 뻗을 때 둘 중 높이가 더 큰 쪽으로 뻗습니다. 이 과정을 반복해서 나오는 직사각형 중 가장 큰 것이 답입니다. 이는 투포인터와 DP로 전처리하여 계산할 수 있습니다.

- 결론적으로 x와 구간이 주어졌을 때, 구간 내의 i에 대해 $A[i] \times (x - i)$를 최소화하는 문제가 되어서, 세그먼트 트리의 각 노드에 컨벡스 헐 트릭 자료구조를 넣어놓는 테크닉으로 풀 수 있습니다.