Omówienie zadań

Mistrzostwa Polski Szkół Średnich w Programowaniu Zespołowym 2024

20 października 2024

Problem A - Liga Mistrzów

Problem A - Liga Mistrzów

Zadanie

Dane są wyniki dwóch meczów pomiędzy dwoma drużynami, gdzie pierwszy mecz był rozgrywany na stadionie pierwszej drużyny, a drugi na stadionie drugiej drużyny.
Określ która drużyna wygrała ten dwumecz, zakładając że w przypadku remisu wygrywa drużyna, która ma więcej strzelonych bramek na wyjeździe.

Problem A - Liga Mistrzów

Zadanie

Dane są wyniki dwóch meczów pomiędzy dwoma drużynami, gdzie pierwszy mecz był rozgrywany na stadionie pierwszej drużyny, a drugi na stadionie drugiej drużyny.
Określ która drużyna wygrała ten dwumecz, zakładając że w przypadku remisu wygrywa drużyna, która ma więcej strzelonych bramek na wyjeździe.

- Sprawdź czy jedna z drużyn strzeliła więcej goli.
- W przypadku remisu sprawdź która strzeliła więcej na wyjeździe.

Problem B - Linia obrony

Problem B - Linia obrony

Zadanie

Mamy dane N punktów z przedziału $[0, D]$.
Każdy z tych punktów ma określony promień zasięgu r_i (w lewo i prawo) oraz koszt c_i, który możemy zapłacić aby zwiekszyć ten promień o 1 (można tę akcję wykonać kilkukrotnie).
Naszym celem jest zwiększenie promieni w taki sposób, żeby żadne zasięgi dwóch punktów nie przecinały się i żeby pokryły one co najmniej cały przedział $[0, D]$ (mogą pokrywać szerszy obszar), a jeśli jest kilka rozwiązań, to aby koszt był najmniejszy.

Problem B - Linia obrony

Obserwacja

Jeżeli ustalimy już promień jednego z punktów, to okazuje się, że promień każdego kolejnego musi mieć konkretną, zależną od niego wartość.

Złożoność rozwiązania, które sprawdza wszystkie możliwe wartości dla pierwszego promienia to $O(N \cdot D)$...

Problem B - Linia obrony

Obserwacja

Jeżeli ustalimy już promień jednego z punktów, to okazuje się, że promień każdego kolejnego musi mieć konkretną, zależną od niego wartość.

Złożoność rozwiązania, które sprawdza wszystkie możliwe wartości dla pierwszego promienia to $O(N \cdot D)$...

... a nawet $O(D)$!

Problem B - Linia obrony

Obserwacja

Jeżeli ustalimy już promień jednego z punktów, to okazuje się, że promień każdego kolejnego musi mieć konkretną, zależną od niego wartość.

Złożoność rozwiązania, które sprawdza wszystkie możliwe wartości dla pierwszego promienia to $O(N \cdot D)$...

... a nawet $O(D)$!

Istnieje również rozwiązanie o złożoności $O(N)$.

Problem C - Uczciwa cena

Problem C - Uczciwa cena

Zadanie

W zadaniu otrzymujemy liczbę początkową S, końcową T, górny limit M, oraz ciąg znaków długości N, składający się z `W`, `P` i `?`.
Rozpoczynamy z liczbą S i patrzymy na kolejne znaki w ciągu. W przypadku `W`, podwajamy aktualną wartość, a w przypadku `P` dzielimy ją bez reszty przez 2. Jeśli natrafimy na `?` to musimy wybrać jedną z tych operacji. Na koniec takiego procesu chcemy otrzymać liczbę T, a także nigdy nie przekroczyć M.

Problem C - Uczciwa cena

Warto od razu pomyśleć o bitowej reprezentacji naszej liczby – wtedy operacje są równoważne usunięciu ostatniego bitu i dopisaniu na końcu 0.

Żeby nasze zadanie było w ogóle możliwe do wykonania, S i T muszą mieć jakiś wspólny prefiks. Co więcej, za tym wspólnym prefiksem w T mogą znaleźć się już tylko bity 0.

Możemy więc policzyć, ile bitów musimy **co najmniej** i **co najwyżej** usunąć z S, żeby móc osiągnąć wartość T.

Problem C - Uczciwa cena

Teraz, gdybyśmy chcieli zasymulować ciąg operacji, wystarczy nam pamiętać:

- jaka długa jest nasza liczba,
- ile bitów na końcu zostało przez nas wyzerowanych.

Zatem, jeśli pominiemy na razie ograniczenie górne M na naszą wartość, możemy napisać program dynamiczny, którego stany to:
`DP[i][b][z]` — czy po przeanalizowaniu pierwszych `i` znaków jesteśmy w stanie otrzymać liczbę o długości `b`, która ma wyzerowane `z` ostatnich bitów.

Zauważmy, że zamiast długości `z` możemy trzymać tylko informację *czy wyzerowaliśmy już wystarczająco bitów*, co redukuje złożoność programu z $O(n \log^2(n))$ do $O(n \log(n))$.

Problem C - Uczciwa cena

Ograniczenie górne w zadaniu wyznacza nam liczbę bitów liczby, której nie możemy przekroczyć.

Jednakże sprawa nie jest aż taka prosta. Zauważmy, że jeśli wyzerujemy w naszej liczbie pewną liczbę końcowych bitów, może się okazać, że będziemy mogli ją wydłużyć o jeden bit więcej.

Dlatego w tej sytuacji potrzebowalibyśmy dodatkowo informacji *czy wyzerowaliśmy już wystarczająco zer, abyśmy mogli osiągnąć długość większą o* 1.

Na szczęście ten 2-bitowy stan nie pogarsza złożoności naszego programu.

Problem C - Uczciwa cena

Okazuje się, że w zadaniu istnieje również poprawne rozwiązanie zachłanne. Polega ono na zachłannym przesuwaniu się tak nisko, jak to tylko możliwe (oczywiście nie zerując przy tym zbyt dużej liczby bitów).
Wtedy, w sytuacji gdy okazuje się, że nasz ciąg zszedłby za nisko, musimy zamienić ostatnią ustawioną przez nas pozycję z `P` na `W`.

Aby sprawdzić czy przy takiej zamianie nie przekraczamy górnego limitu, możemy użyć drzewa przedziałowego. Ale istnieje też prostszy sposób – możemy nie przejmować się limitem górnym w trakcie, gdyż nie jesteśmy w stanie nic poradzić na jego przekroczenie. Zamiast tego możemy po prostu sprawdzić na końcu czy nasza sekwencja jest poprawna.

Problem D - Numery na koszulkach

Problem D - Numery na koszulkach

Treść

Mamy dane drzewo. Mamy ponumerować jego wierzchołki w taki sposób by żadne dwa połączone krawędzią nie różniły się o 1 (modulo n).

Problem D - Numery na koszulkach

Obserwacja 1

Zauważmy, że zadanie można zinterpretować jako dodanie do grafu cyklu prostego długości n, takiego że nie korzysta on z żadnej krawędzi w oryginalnym grafie (krawędzie tego cyklu odpowiadają parom wierzchołków, które nie mogą być sąsiadami).

Problem D - Numery na koszulkach

Obserwacja 1

Zauważmy, że zadanie można zinterpretować jako dodanie do grafu cyklu prostego długości n, takiego że nie korzysta on z żadnej krawędzi w oryginalnym grafie (krawędzie tego cyklu odpowiadają parom wierzchołków, które nie mogą być sąsiadami).

Obserwacja 2

Zauważmy, że drzewo jest grafem dwudzielnym. Możemy pokolorować wierzchołki dwoma kolorami tak, aby żadne dwa o tym samym kolorze ze sobą nie sąsiadowały. Oznacza to, że wierzchołki o tym samym kolorze mogą bez problemu być swoimi sąsiadami w cyklu, który chcemy dodać

Problem D - Numery na koszulkach

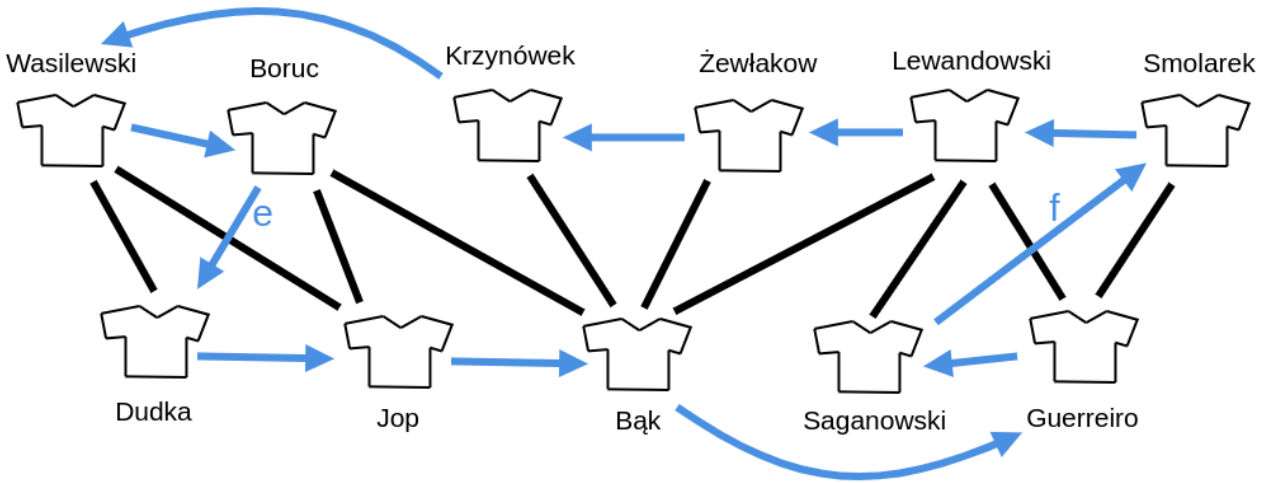
Wniosek

Do rozłożenia cyklu wystarczą nam zatem dwie krawędzi e i f łączące wierzchołki o różnych kolorach, takie że e i f nie występują w drzewie.

Problem D - Numery na koszulkach

Wniosek

Do rozłożenia cyklu wystarczą nam zatem dwie krawędzi e i f łączące wierzchołki o różnych kolorach, takie że e i f nie występują w drzewie. Możemy wtedy odwiedzić wszystkie wierzchołki jednego koloru, a następnie przedostać się na drugą stronę krawędzią e, odwiedzić całą drugą stronę i domknąć cykl krawędzią f.



Problem D - Numery na koszulkach

Konieczność

Łatwo zauważyć, że warunek takich dwóch krawędzi jest nie tylko wystarczalny ale też konieczny, ponieważ rozcięcie cyklu wymaga przynajmniej dwóch cięć.

Problem D - Numery na koszulkach

Konieczność

Łatwo zauważyć, że warunek takich dwóch krawędzi jest nie tylko wystarczalny ale też konieczny, ponieważ rozcięcie cyklu wymaga przynajmniej dwóch cięć.

Potencjalne problemy

Jeżeli krawędzie e i f miałyby koniec wspólny, to wchodząc do niego nie moglibyśmy odwiedzić wszystkich wierzchołków o tym samym kolorze. Pozostaje przypadek, że byłby jedynym wierzchołkiem o jakimś kolorze, ale wtedy rozwiązanie nie istnieje.

Problem D - Numery na koszulkach

Jak znaleźć dwie krawędzie o różnych końcach, które nie występują w grafie? Znalezienie jednej jest możliwe w czasie $O(m \cdot log(m))$ (iterując się po wszystkich parach wierzchołków i wypisując pierwszą z nich, której nie ma w grafie).

Problem D - Numery na koszulkach

Jak znaleźć dwie krawędzie o różnych końcach, które nie występują w grafie? Znalezienie jednej jest możliwe w czasie $O(m \cdot log(m))$ (iterując się po wszystkich parach wierzchołków i wypisując pierwszą z nich, której nie ma w grafie).

Każdy kolor ma przynajmniej 3 wierzchołki

Znajdźmy krawędź e i usuńmy jej końce z dalszych rozważań. Zauważmy, że wśród pozostałych wierzchołków każdy kolor ma przynajmniej dwa wierzchołki, czyli dobra krawędź f istnieje (szukamy jej tak samo wśród pozostałych).

Problem D - Numery na koszulkach

Jak znaleźć dwie krawędzie o różnych końcach, które nie występują w grafie? Znalezienie jednej jest możliwe w czasie $O(m \cdot log(m))$ (iterując się po wszystkich parach wierzchołków i wypisując pierwszą z nich, której nie ma w grafie).

Każdy kolor ma przynajmniej 3 wierzchołki

Znajdźmy krawędź e i usuńmy jej końce z dalszych rozważań. Zauważmy, że wśród pozostałych wierzchołków każdy kolor ma przynajmniej dwa wierzchołki, czyli dobra krawędź f istnieje (szukamy jej tak samo wśród pozostałych).

Jeden kolor ma dokładnie 2 wierzchołki

Wtedy wiemy, że jeden z nich jest końcem e a drugi końcem f, więc dla każdego kandydata na e łatwo sprawdzić, czy istnieje jakiś poprawny kandydat na f.

D - Numery na koszulkach - Rozwiązania alternatywne

Rozwiązanie zrandomizowane

Istnieje przynajmniej jedna zrandomizowana wersja powyższego rozwiązania, ale nie wydaje się ona prostsza i posiada parę miejsc, gdzie łatwo o błędy w analizie prawdopodobieństwa.

D - Numery na koszulkach - Rozwiązania alternatywne

Rozwiązanie zrandomizowane

Istnieje przynajmniej jedna zrandomizowana wersja powyższego rozwiązania, ale nie wydaje się ona prostsza i posiada parę miejsc, gdzie łatwo o błędy w analizie prawdopodobieństwa.

Rozwiązanie z centroidem

Przeglądajmy kolejno poddrzewa po ukorzenieniu w centroidzie (od największych), przypisując numery nieparzyste rosnąco albo parzyste malejąco, (wybierając z aktualnie większego zbioru). Może się okazać, że na końcu zostanie nam jakieś poddrzewo (+ centroid), którym trzeba będzie przypisać zarówno parzyste jak i nieparzyste numery, ale zauważmy, że zużyliśmy przynajmniej połowę parzystych i nieparzystych (więc nie ma problemu z $n/2$ i $n/2+1$). Tutaj niestety również jest parę przypadków brzegowych do rozważenia.

Problem E - Taktyczna rozgrywka

Problem E - Taktyczna rozgrywka

Zadanie

Mamy daną planszę $n \times m$. Niektóre jej pola są zablokowane – dokładniej w i-tej kolumnie zablokowane są pola $(p_i \cdot k, i)$, gdzie p_i dane na wejściu jest liczbą pierwszą a $k \in \mathbb{Z}_+$. Chcemy znaleźć, dla Q pól z zapytań koszt dotarcia na do ostatniej kolumny, gdzie:

- możemy przejść z (x, y) do $(x, y + 1)$ kosztem 0
- możemy przejść z (x, y) do $(x \pm d, y + 1)$ kosztem d

Oczywiście nie możemy wchodzić na zablokowane pola

Problem E - Taktyczna rozgrywka

Zadanie

Mamy daną planszę $n \times m$. Niektóre jej pola są zablokowane – dokładniej w i-tej kolumnie zablokowane są pola $(p_i \cdot k, i)$, gdzie p_i dane na wejściu jest liczbą pierwszą a $k \in \mathbb{Z}_+$. Chcemy znaleźć, dla Q pól z zapytań koszt dotarcia na do ostatniej kolumny, gdzie:

- możemy przejść z (x, y) do $(x, y + 1)$ kosztem 0
- możemy przejść z (x, y) do $(x \pm d, y + 1)$ kosztem d

Oczywiście nie możemy wchodzić na zablokowane pola

Ograniczenia

$n, m \leqslant 10^5, q \leqslant 10^6$

Problem E - Taktyczna rozgrywka

Definicja

Niech S_d to zbiór pól, których odległość do końca jest co najwyżej d

Problem E - Taktyczna rozgrywka

Definicja

Niech S_d to zbiór pól, których odległość do końca jest co najwyżej d

Obserwacja

Odległość pól do końca w danym wierszu jest malejąca.

Problem E - Taktyczna rozgrywka

Definicja

Niech S_d to zbiór pól, których odległość do końca jest co najwyżej d

Obserwacja

Odległość pól do końca w danym wierszu jest malejąca.

Obserwacja

Istnieje optymalne rozwiązanie, które w każdym kroku zmienia wiersz o maksymalnie 1.

Problem E - Taktyczna rozgrywka

Definicja

Niech S_d to zbiór pól, których odległość do końca jest co najwyżej d

Obserwacja

Odległość pól do końca w danym wierszu jest malejąca.

Obserwacja

Istnieje optymalne rozwiązanie, które w każdym kroku zmienia wiersz o maksymalnie 1.

Obie obserwacje dowodzi się podobnie jak algorytmy zachłanne – rozważamy optymalny ciąg ruchów, następnie modyfikujemy go żeby był bliżej zachowania danej własności.

Problem E - Taktyczna rozgrywka

S_0 to po prostu pola za ostatnimi przeszkodami w każdym wierszu.

Problem E - Taktyczna rozgrywka

S_0 to po prostu pola za ostatnimi przeszkodami w każdym wierszu.

Wyliczanie S_{d+1} za pomocą S_d

Niech $(x_1, y-1), (x_2, y+1) \in S_d$ będą pierwszymi polami w rzędach $y-1, y+1$.
Wtedy możemy z nich przeskoczyć kosztem 1 do $(min(x_1, x_2) - 1, y)$, zatem to pole $\in S_{d+1}$. Oznaczając przez (x, y) takie pole, że pomiędzy (x, y) a $(min(x_1, x_2) - 1, y)$ nie ma przeszkód, dostajemy, że (x, y) to pierwsze pole w tym rzędzie $\in S_{d+1}$. Dodatkowo należy uwzględnić przypadek, gdy już wcześniej było ono w S_d.

Problem E - Taktyczna rozgrywka

S_0 to po prostu pola za ostatnimi przeszkodami w każdym wierszu.

Wyliczanie S_{d+1} za pomocą S_d

Niech $(x_1, y-1), (x_2, y+1) \in S_d$ będą pierwszymi polami w rzędach $y-1, y+1$.
Wtedy możemy z nich przeskoczyć kosztem 1 do $(min(x_1, x_2) - 1, y)$, zatem to pole $\in S_{d+1}$. Oznaczając przez (x, y) takie pole, że pomiędzy (x, y) a $(min(x_1, x_2) - 1, y)$ nie ma przeszkód, dostajemy, że (x, y) to pierwsze pole w tym rzędzie $\in S_{d+1}$. Dodatkowo należy uwzględnić przypadek, gdy już wcześniej było ono w S_d.

Zajmuje to czas liniowy od liczby wierszy, które mają jeszcze nieodwiedzone pola. Łącznie zatem, jest to suma odległości pierwszych pól w rzędach od końca.

Problem E - Taktyczna rozgrywka

Po posortowaniu zapytań, na każde poświęcamy stały czas w zamiataniu. Zatem, wystarczy tylko policzyć, jaka jest suma odległości pól w pierwszej kolumnie do końca

Analiza złożoności

Załóżmy, że $m = n$. Szacowanie górne będzie korzystać z następujących ścieżek:

- idziemy do najbliższej liczby pierwszej, która jest brana w co najwyżej Δ wierszach
- dla tej liczby pierwszej idziemy do końca

dla pewnej stałej Δ dobranej później.

Problem E - Taktyczna rozgrywka

Analiza złożoności

Po pewnych obliczeniach (dostępnych w omówieniu) dochodzimy do oszacowania przez

$$n\Delta + G^2 \frac{n^2}{2\Delta^2},$$

gdzie G to największa luka w liczbach pierwszych.

Problem E - Taktyczna rozgrywka

Analiza złożoności

Po pewnych obliczeniach (dostępnych w omówieniu) dochodzimy do oszacowania przez

$$n\Delta + G^2 \frac{n^2}{2\Delta^2},$$

gdzie G to największa luka w liczbach pierwszych. Zatem, należy dobrać odpowiednie Δ, żeby ta funkcja była minimalna, dokładniej $n\Delta = G^2 \frac{n^2}{\Delta^2}$, co da:

$$\frac{3}{2} n^{\frac{4}{3}} G^{\frac{2}{3}}.$$

Problem E - Taktyczna rozgrywka

Analiza złożoności

Po pewnych obliczeniach (dostępnych w omówieniu) dochodzimy do oszacowania przez

$$n\Delta + G^2 \frac{n^2}{2\Delta^2},$$

gdzie G to największa luka w liczbach pierwszych. Zatem, należy dobrać odpowiednie Δ, żeby ta funkcja była minimalna, dokładniej $n\Delta = G^2 \frac{n^2}{\Delta^2}$, co da:

$$\frac{3}{2} n^{\frac{4}{3}} G^{\frac{2}{3}}.$$

Przy czym wiadomo, że $G = \mathcal{O}(n^{0.525})$, czyli mamy złożoność $\mathcal{O}(n^{1.68} + Q)$.

Problem E - Taktyczna rozgrywka

Analiza złożoności

Po pewnych obliczeniach (dostępnych w omówieniu) dochodzimy do oszacowania przez

$$n\Delta + G^2 \frac{n^2}{2\Delta^2},$$

gdzie G to największa luka w liczbach pierwszych. Zatem, należy dobrać odpowiednie Δ, żeby ta funkcja była minimalna, dokładniej $n\Delta = G^2 \frac{n^2}{\Delta^2}$, co da:

$$\frac{3}{2} n^{\frac{4}{3}} G^{\frac{2}{3}}.$$

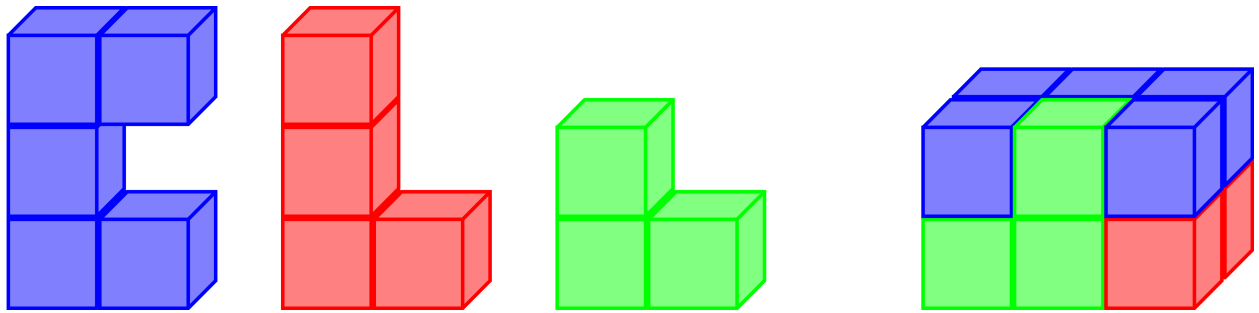
Przy czym wiadomo, że $G = \mathcal{O}(n^{0.525})$, czyli mamy złożoność $\mathcal{O}(n^{1.68} + Q)$. Numerycznie możemy oszacować lepiej: $G = 72$, co daje łącznie $3.7 \cdot 10^7$ z możliwościami poprawy oszacowania.

Problem F - Klocki

Problem F - Klocki

Zadanie

Mamy daną figurę składającą się z sześcianów oraz kilka klocków również składających się z takich sześcianów.
Celem zadania jest określenie czy z tych klocków da się ułożyć podaną figurę, a jeśli tak, to opisanie który klocek będzie gdzie.

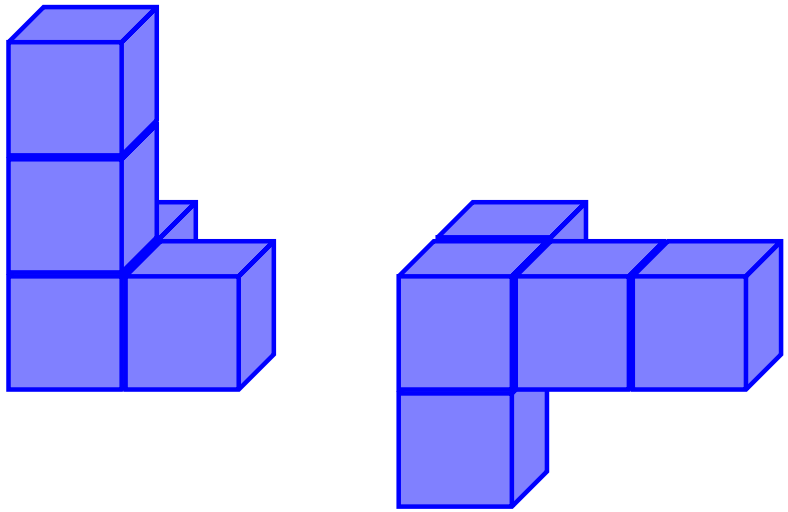


Problem F - Klocki

Po pierwsze, trzeba zaimplementować obracanie klocków.

Na płaszczyźnie zamieniamy $(x, y) \to (y, -x)$.

W trzecim wymiarze (obróćmy klocka wokół osi OZ) zamieniamy $(x, y, z) \to (y, -x, z)$.



Problem F - Klocki

Wszystkich możliwych obrotów każdego klocka jest do 24.

Można wygenerować wszystkie 64 obrotów, a potem wyrzucić powtórzenia.

Potrzebna jest normalizacja (czyli przesunięcie klocka w pewien charakterystyczny sposób do punktu $(0, 0, 0)$).

Teraz można napisać rozwiązanie używające backtrackingu, ale niestety jest ono jeszcze za wolne...

Problem F - Klocki

Użyjmy masek bitowych do optymalizacji.

Ponumerujmy wszystkie sześciany układanej figury.

Teraz dla każdego klocka możemy sprawdzić wszystki jego obroty i przesunięcia, a jeśli któreś pokrywa sie z figurą, to dodajemy maskę pokrytych sześcianów do puli pasujących masek.

Teraz mozemy napisać rozwiązanie używające backtrackingu, ale zapamiętujące odwiedzone już stany.

Kolejno dla każdego klocka, dla każdej maski pokrytych już klocków oraz każdej maski możliwej do pokrycia wywołujemy funkcję rekurencyjną – to rozwiązanie zawiera bardzo dużo stanów, ale też bardzo dużą część z nich pomija.

Problem G - Okno transferowe

Problem G - Okno transferowe

Zadanie

W zadaniu mamy dane N ofert na rynku transferowym, z czego każda jest parametryzowana przez:

- prawdopodobieństwo, że się pojawi na rynku,
- kiedy pojawi się na rynku,
- do kiedy trzeba podjąć decyzję o jej przyjęciu,
- ile można zarobić.

Po wybraniu jednej oferty nie możemy rozważać innej.
Jaka jest oczekiwana wartość naszych zarobków, jeśli będziemy podejmować decyzje optymalnie?

Problem G - Okno transferowe

Zacznijmy od nieefektywnego rozwiązania za pomocą programowania dynamicznego. Oznaczmy poprzez $dp[t][S]$ wartość oczekiwaną optymalnej strategii wybierania ofert, jeśli nie rozważyliśmy jeszcze przedziałów o $s_i >= t$, a zbiór S oznacza istniejące przedziały spełniające $s_i \leqslant t \leqslant e_i$.

Takie programowanie dynamiczne można w prosty sposób liczyć:

- Jeśli w chwili t dodajemy nową ofertę r, to $dp[t][S] = p_r \cdot dp[t+1][S \cup \{r\}] + (1 - p_r) \cdot dp[t+1][S]$;
- Jeśli w chwili t kończy się oferta $r \in S$, to $dp[t][S] = \max\{dp[t+1][S \setminus \{r\}], v_r\}$;
- Jeśli w chwili t kończy się oferta $r \notin S$, to $dp[t][S] = dp[t+1][S]$.

Problem G - Okno transferowe

Kluczowa obserwacja jest taka, że wystarczy rozważać zbiory S spełniające $|S| \leqslant 1$.

Obserwacja ta pozwala nam na optymalizację naszego dynamika do złożoności $\mathcal{O}(n^2)$, co nie jest jeszcze wystarczające do zaakceptowania zadania.

Problem G - Okno transferowe

Aby zoptymalizować powyższe rozwiązanie, rozpiszmy przejścia dynamika z $|S| \leqslant 1$ w nieco inny sposób:

- Jeśli w chwili t dodajemy nową ofertę r, to dla każdej istniejącej oferty $s \neq r$ zapisujemy $dp[t][s] = \max\{dp[t+1][s], (1-p_r) \cdot dp[t+1][s] + p_r \cdot dp[t+1][r]\}$, a stan $dp[t][r]$ usuwamy;
- Jeśli w chwili t usuwamy ofertę r, to dodajemy stan $dp[t][r] = \max\{dp[t+1][0], v_r\}$, a pozostałe stany pozostają bez zmian.

Problem G - Okno transferowe

Powyższy wzór jesteśmy w stanie liczyć dość szybko na dowolnej strukturze, która pozwala nam na następujące operację:

- Usuń element o wartości v;
- Dodaj element o wartości v;
- Zmodyfikuj wszystkie elementy o wartości $\leqslant v$ wzorem $\alpha \cdot v + \beta$. Zauważmy, że ta operacja nie zmienia kolejności sortowania elementów.

Do zaimplementowania tych operacji możemy użyć dowolnego zbalansowanego drzewa binarnego, przykładowo treapa, co daje nam złożoność $\mathcal{O}(\log n)$.

Zadanie to można rozwiązać jednak prościej, za pomocą struktury pierwiastkowej. Idea jest taka, aby utrzymywać aktywne stany, posortowane po wartościach dynamika, w kubełkach o długości pierwiastkowej.

Problem H - Hiszpański komentarz

Problem H - Hiszpański komentarz

Zadanie

Analizując zadany komentarz mamy policzyć ile goli padło w meczu wiedząc, że:

- po każdym golu pada słowo `GOL`,
- wielokrotne wystąpienie słowa `GOL` bezpośrednio po sobie oznacza jednego gola,
- litera `O` wewnątrz słowa może występować wielokrotnie,
- tekst zawiera małe i duże litery.

Aby rozwiązać zadanie należało:

- zamienić wszystkie litery na małe,
- usunąć wielokrotne litery `O`,
- zliczyć ile razy występuje słowo `gol`, ale tak aby bezpośrednio za nim nie występowało słowo `gol`.

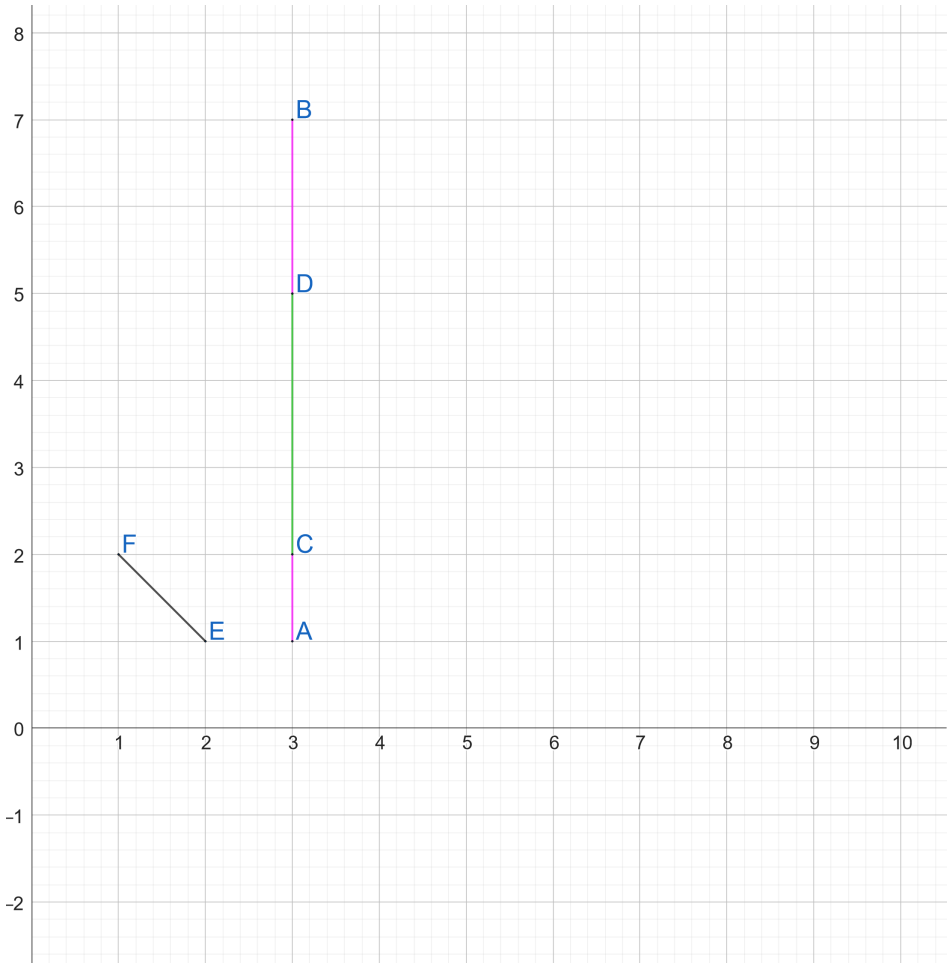
Problem I - Precyzyjny strzał

Problem I - Precyzyjny strzał

Zadanie

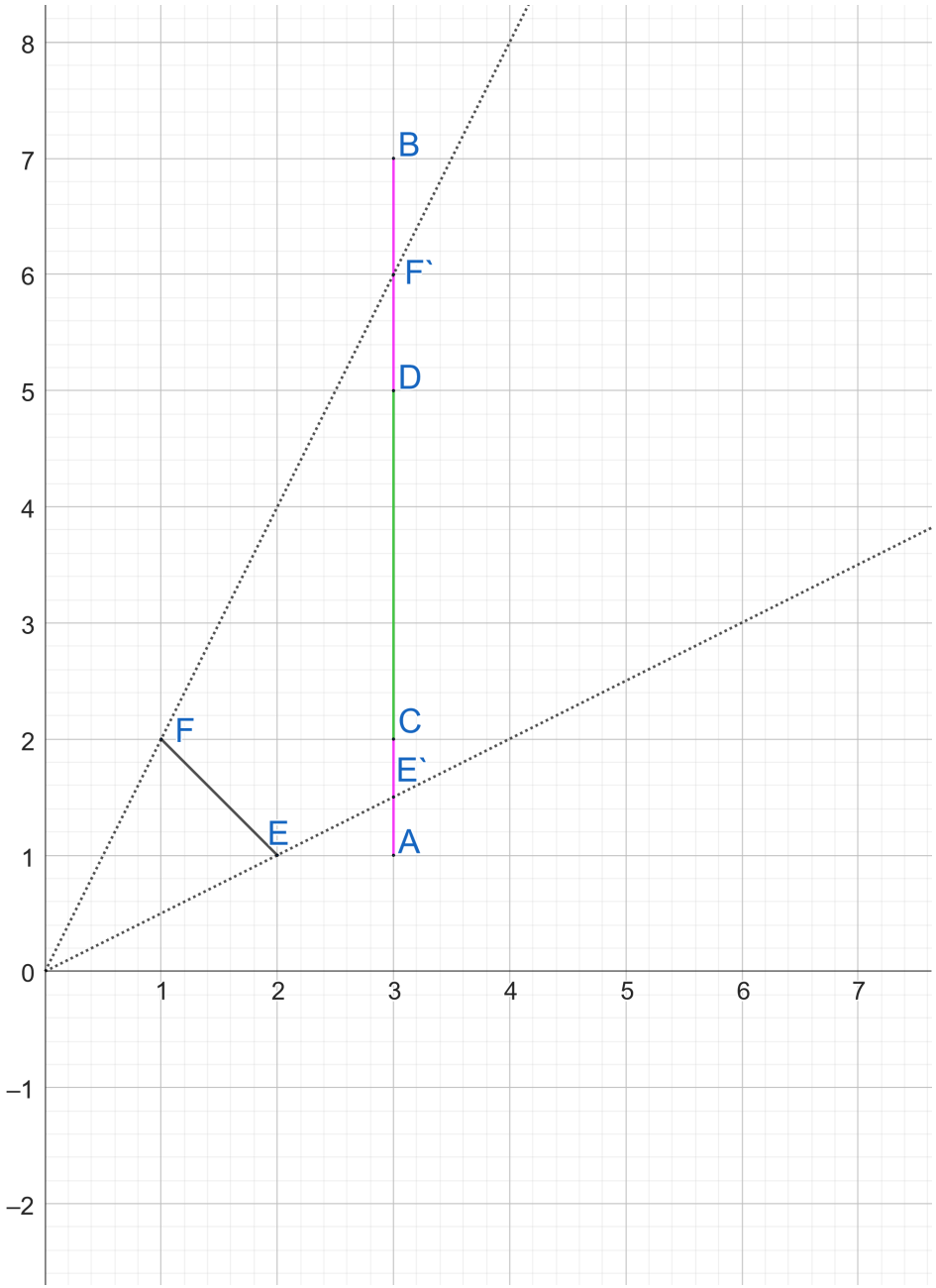
W zadaniu dane są trzy odcinki: AB równoległy do osi OY, CD zawarty wewnątrz AB oraz EF, który znajduje się na lewo od AB oraz na prawo od osi OY. Wszystkie współrzędne są liczbami całkowitymi. Należy sprawdzić, czy istnieje półprosta zaczynająca się w $(0, 0)$ i przecinająca tylko odcinek AB.

Problem I - Precyzyjny strzał



Problem I - Precyzyjny strzał - rozwiązanie

Zauważmy, że odcinek CD wydziela z AB dwa odcinki (być może zdegenerowane do punktu): AC oraz DB. Wystarczy więc sprawdzić, czy istnieje półprosta przechodząca przez jeden z nich, ale nie przez EF i nie przez punkty C ani D.
Rozważmy półprostą przechodząca przez AC, drugi przypadek jest analogiczny. Kiedy taka półprosta może istnieć? Wtedy, gdy odcinek EF nie "zasłania"całego odcinka AC.
Żeby to sprawdzić, możemy rzutować odcinek EF na prostą zawierającą odcinek AC i sprawdzić czy punkt E' (rzut punktu E) jest powyżej A lub F' jest poniżej C:


B
F`
D
C
E`
A
F
E
8
7
6
5
4
3
2
1
0
−1
−2
1
2
3
4
5
6
7

Problem I - Precyzyjny strzał - rozwiązanie

Współrzędną y punktu E' można obliczyć w następujący sposób:

$$E'_y = \frac{E_y}{E_x} \cdot A_x$$

Interesujące nas porównanie punktów E' oraz A ma zatem postać:

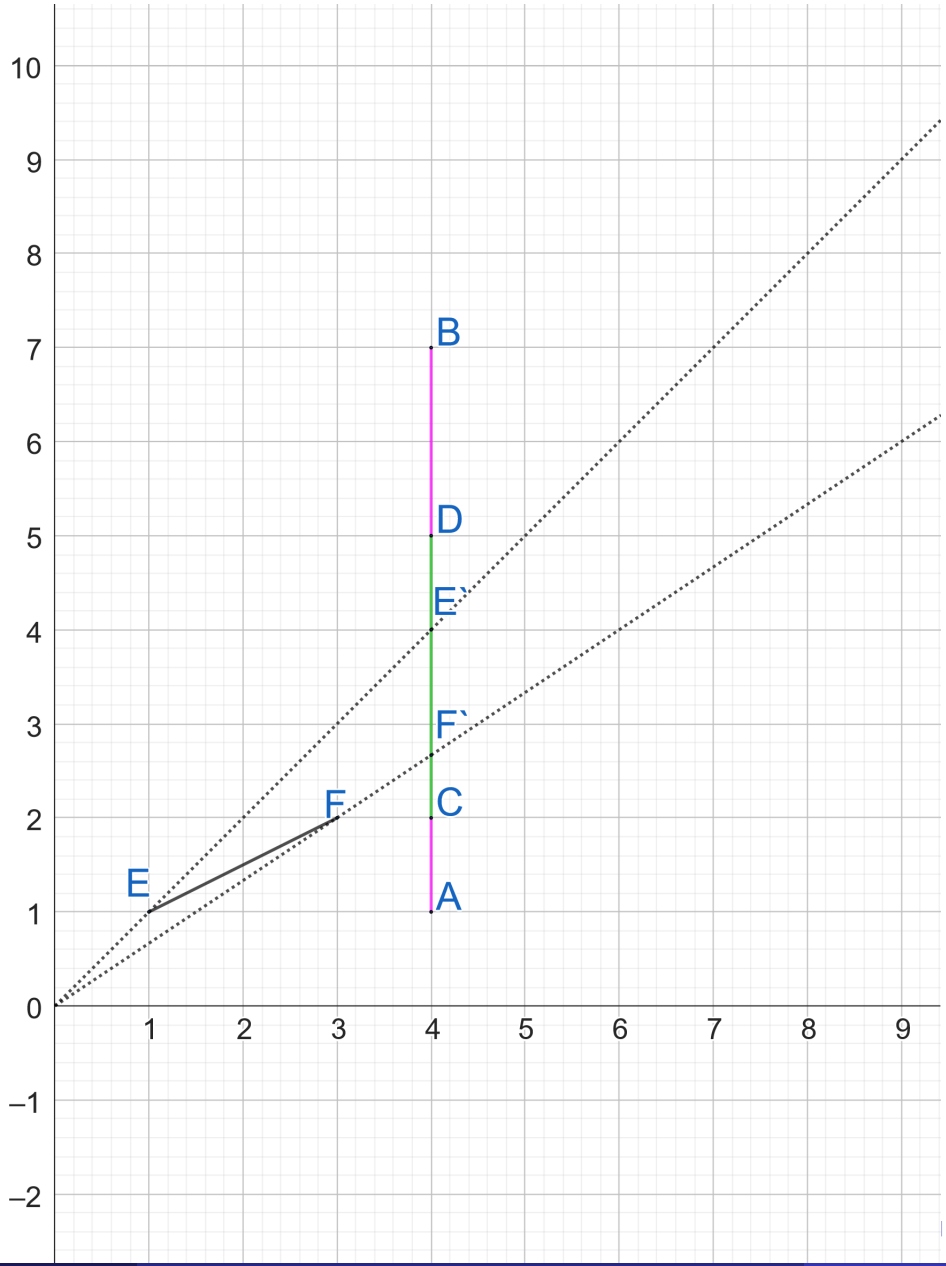
$$E_y \cdot A_x > A_y \cdot E_x$$

Porównianie F' z C jest analogicznie

Problem I - Precyzyjny strzał - uwagi

- Trzeba uważać na overflow przy wykonaniu mnożenia (najlepiej użyć większego typu, np. `long long`).
- Jeżeli do obliczeń używasz liczb zmiennoprzecinkowych albo funkcji trygonometrycznych, pamiętaj o korzystaniu z `long double` (i np. `atan2l`), inaczej precyzja będzie niewystarczająca.
- Po rzucie odcinka EF, punkt F' może być niżej niż E', wtedy trzeba porównać pozycje F' z A oraz E' z C.

Problem I - Precyzyjny strzał - uwagi



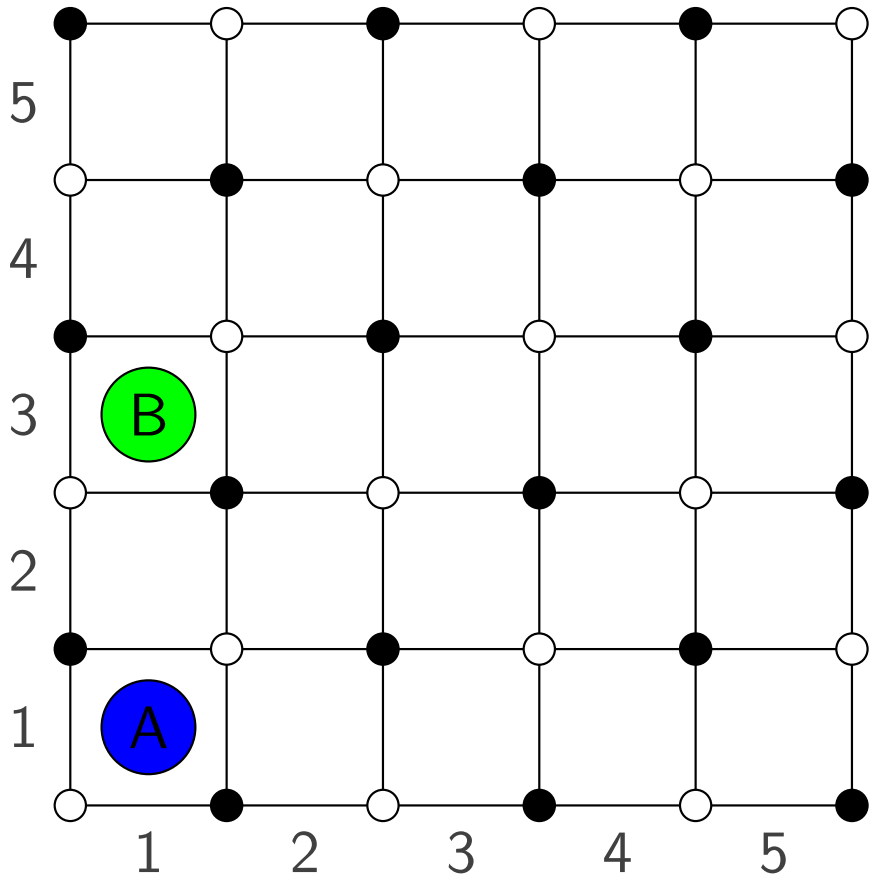
Problem J - Trening z pachołkami

Problem J - Trening z pachołkami

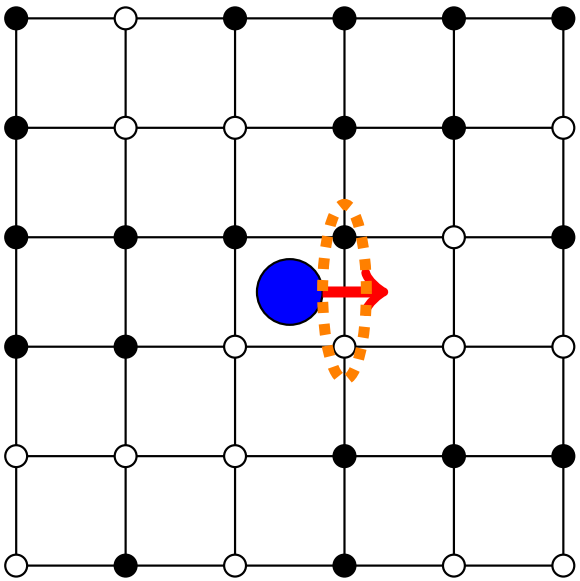
Zadanie

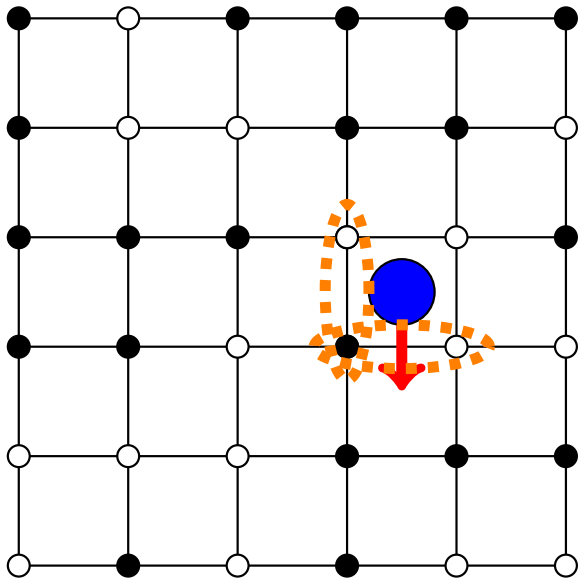
Na dwóch polach nieskończonej planszy stoją pionki. Dodatkowo, na każdym rogu czterech pól stoi żeton jednego z dwóch kolorów, czarny lub biały. Ruch pionkiem polega na przesunięciu go do sąsiedniego pola i zamienieniu żetonów na wspólnym boku. Można go zrobić tylko wtedy gdy kolory tych żetonów są różne. Na początku żetony ustawione są w szachownicę. Trzeba zamienić pionki miejscami.

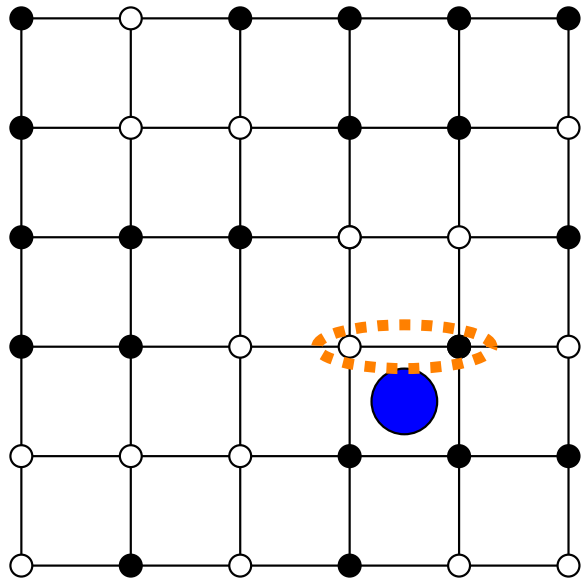
Problem J - Trening z pachołkami



Problem J - Trening z pachołkami







Problem J - Trening z pachołkami

Rozwiązanie istnieje gdy początkowo pionki stoją na polach (x_a, y_a), (x_b, y_b) takich, że nie zachodzi $|x_a - x_b| = |y_a - y_b| = 1$ (to znaczy, gdy pola nie sąsiadują rogami).

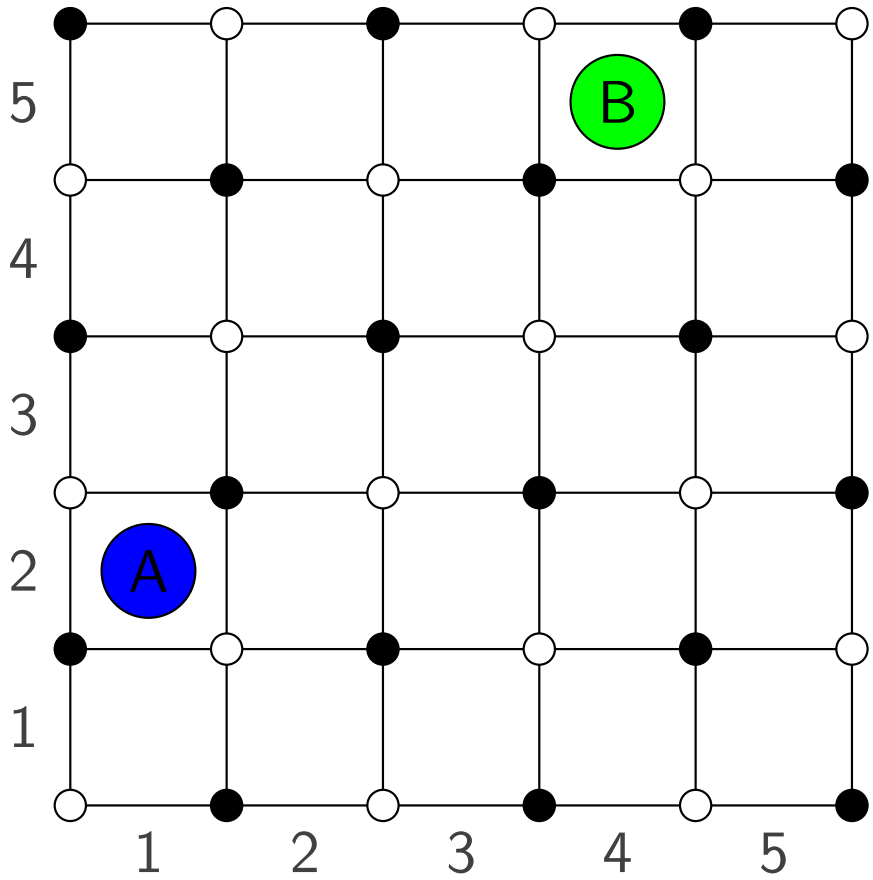
Problem J - Trening z pachołkami

Rozwiązanie istnieje gdy początkowo pionki stoją na polach (x_a, y_a), (x_b, y_b) takich, że nie zachodzi $|x_a - x_b| = |y_a - y_b| = 1$ (to znaczy, gdy pola nie sąsiadują rogami).

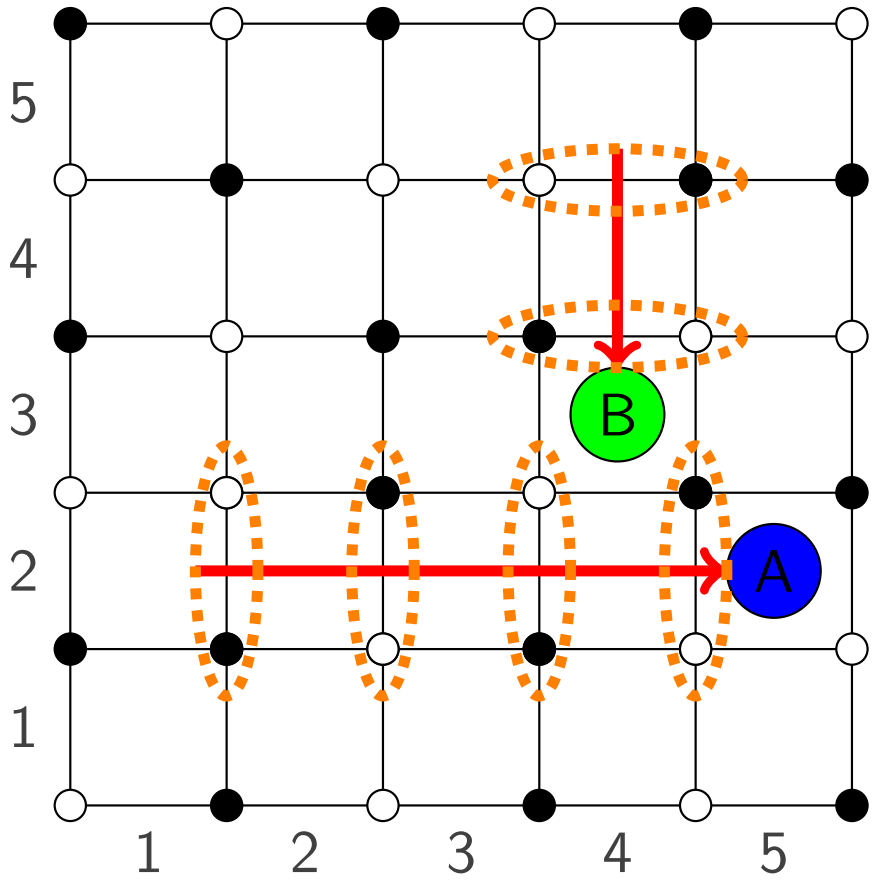
Obserwacja

Możemy ustawić pionki na jednym polu a potem odwrócić wykonane ruchy.

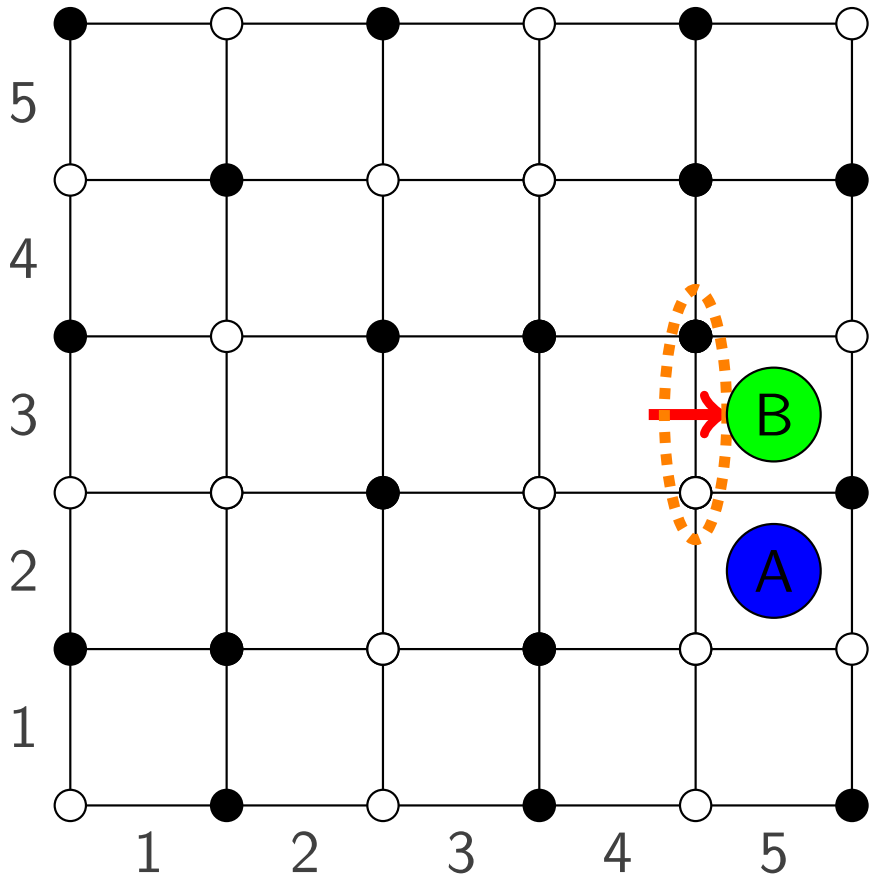
Problem J - Trening z pachołkami



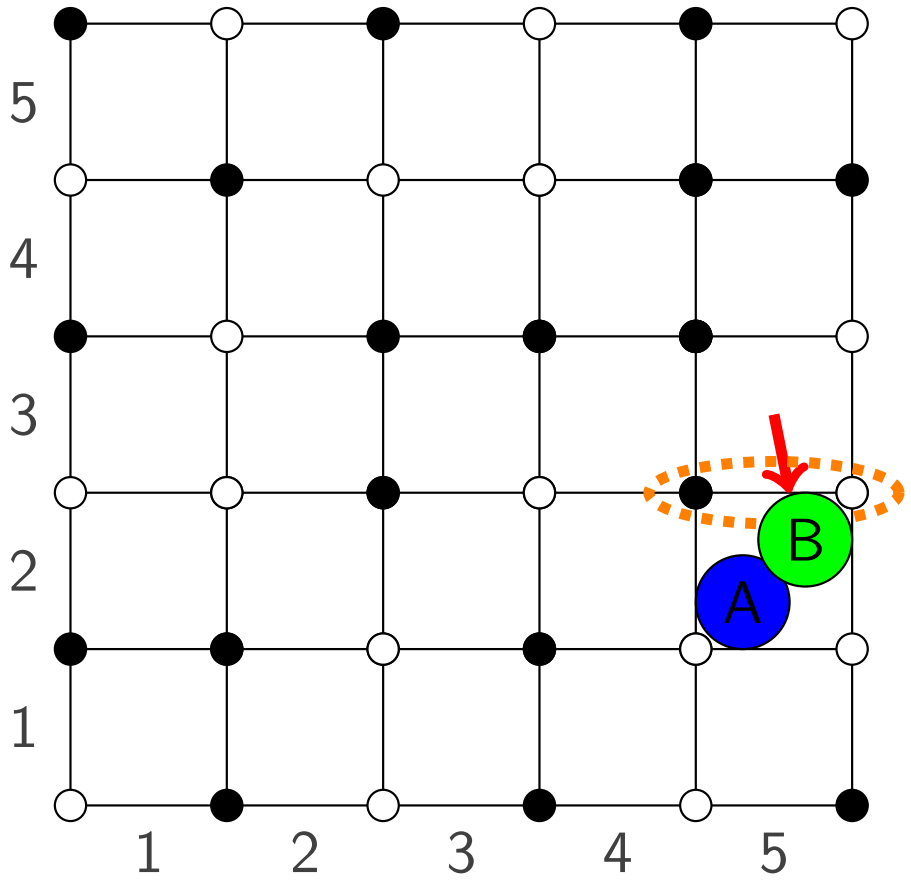
Problem J - Trening z pachołkami



Problem J - Trening z pachołkami



Problem J - Trening z pachołkami



Problem K - Piłkarskie makao

Problem K - Piłkarskie makao

Zadanie

W zadaniu mamy dane drzewo o n wierzchołkach przy czym i-ty z nich jest etykietowany liczbą całkowitą d_i. Dla każdego wierzchołka v należy obliczyć iloczyn funkcji J obliczonej dla każdego poddrzewa powstałego z usunięcia wierzchołka v z grafu:

$$J = \sum_{t=1}^{\infty} t \cdot W_{\mathrm{cnt}_t}$$

Gdzie współczynniki W są dane (przy czym $W_0 = 0$), a cnt_t to liczba etykiet w danym poddrzewie, które są wielokrotnościami t.

Problem K - Piłkarskie makao

Obserwacja

Jeśli mamy zbiór S wierzchołków, dla którego mamy policzoną funkcję z zadania to w $\mathcal{O}(\text{liczba dzielników } d_v)$ możemy policzyć jej wartość dla $S \cup v$ lub $S \setminus v$.

Problem K - Piłkarskie makao

Obserwacja

Jeśli mamy zbiór S wierzchołków, dla którego mamy policzoną funkcję z zadania to w $\mathcal{O}(\text{liczba dzielników } d_v)$ możemy policzyć jej wartość dla $S \cup v$ lub $S \setminus v$.
Zatem możemy utrzymywać powyższą strukturę i wykonując na niej pewne operacje, chcemy, żeby dla każdego wierzchołka istniał moment, że S jest równy poddrzewu v. Równocześnie utrzymując dopełnienie S będziemy mieli wszystkie informacje potrzebne do odzyskania wyniku.

Problem K - Piłkarskie makao

Algorytm

Możemy skorzystać z kolejności DFS:

Problem K - Piłkarskie makao

Algorytm

Możemy skorzystać z kolejności DFS:

1. Rekursywnie przetwarzamy wszystkie dzieci v z wyjątkiem jednego (y). Po przetworzeniu każdego poddrzewa, przechodzimy je jeszcze raz usuwając wszystkie wierzchołki ze struktury danych.

Problem K - Piłkarskie makao

Algorytm

Możemy skorzystać z kolejności DFS:

1. Rekursywnie przetwarzamy wszystkie dzieci v z wyjątkiem jednego (y). Po przetworzeniu każdego poddrzewa, przechodzimy je jeszcze raz usuwając wszystkie wierzchołki ze struktury danych.
2. Rekursywnie przetwarzamy dziecko y.

Problem K - Piłkarskie makao

Algorytm

Możemy skorzystać z kolejności DFS:

1. Rekursywnie przetwarzamy wszystkie dzieci v z wyjątkiem jednego (y). Po przetworzeniu każdego poddrzewa, przechodzimy je jeszcze raz usuwając wszystkie wierzchołki ze struktury danych.
2. Rekursywnie przetwarzamy dziecko y.
3. Dodajemy v do struktury oraz wszystkie wierzchołki z poddrzew dzieci v z pierwszego punktu. Wtedy struktura zawiera wszystkie wierzchołki z poddrzewa v, więc możemy zapisać wartość J dla poddrzewa v.

Problem K - Piłkarskie makao

Algorytm

Możemy skorzystać z kolejności DFS:

1. Rekursywnie przetwarzamy wszystkie dzieci v z wyjątkiem jednego (y). Po przetworzeniu każdego poddrzewa, przechodzimy je jeszcze raz usuwając wszystkie wierzchołki ze struktury danych.
2. Rekursywnie przetwarzamy dziecko y.
3. Dodajemy v do struktury oraz wszystkie wierzchołki z poddrzew dzieci v z pierwszego punktu. Wtedy struktura zawiera wszystkie wierzchołki z poddrzewa v, więc możemy zapisać wartość J dla poddrzewa v.

Nie chcemy jednak wykonywać za dużo operacji na strukturze, gdyż każdy wierzchołek może mieć dużo dzielników.

Problem K - Piłkarskie makao

Mniejszy do większego

Każdy wierzchołek u zostanie tyle razy wrzucony i wyciągnięty ze struktury ile jest wierzchołków v na ścieżce od niego do korzenia, że y (dla v) nie jest przodkiem u.

Problem K - Piłkarskie makao

Mniejszy do większego

Każdy wierzchołek u zostanie tyle razy wrzucony i wyciągnięty ze struktury ile jest wierzchołków v na ścieżce od niego do korzenia, że y (dla v) nie jest przodkiem u.
Zatem, niech y będzie dzieckiem v o największym rozmiarze poddrzewa.

Problem K - Piłkarskie makao

Mniejszy do większego

Każdy wierzchołek u zostanie tyle razy wrzucony i wyciągnięty ze struktury ile jest wierzchołków v na ścieżce od niego do korzenia, że y (dla v) nie jest przodkiem u.
Zatem, niech y będzie dzieckiem v o największym rozmiarze poddrzewa. Wówczas, każde u jeśli nie jest potomkiem y to rozmiar y jest co najmniej równy rozmiarowi u' (brat y, przodek u). Ale to znaczy że rozmiar poddrzewa v jest co najmniej dwukrotnie większy od rozmiaru poddrzewa u'. Ale taka sytuacja może nastąpić dla konkretnego y tylko $\log(n)$ razy.

Problem K - Piłkarskie makao

Mniejszy do większego

Każdy wierzchołek u zostanie tyle razy wrzucony i wyciągnięty ze struktury ile jest wierzchołków v na ścieżce od niego do korzenia, że y (dla v) nie jest przodkiem u.
Zatem, niech y będzie dzieckiem v o największym rozmiarze poddrzewa. Wówczas, każde u jeśli nie jest potomkiem y to rozmiar y jest co najmniej równy rozmiarowi u' (brat y, przodek u). Ale to znaczy że rozmiar poddrzewa v jest co najmniej dwukrotnie większy od rozmiaru poddrzewa u'. Ale taka sytuacja może nastąpić dla konkretnego y tylko $\log(n)$ razy.

Złożoność

$\mathcal{O}(S \log n)$, gdzie $S \leqslant 240n$ jest sumą dzielników wszystkich liczb d_v.

Problem K - Piłkarskie makao

Bonus - rozwiązanie bez czynnika log

Dla każdego d skompresujmy drzewo do wielokrotności d. Każda z nich wyznacza pewne sumy prefiksowe, które można potem rozpropagować (na przykład za pomocą techniki o nazwie rerooting) na początkowe drzewo. Pierwszy krok zajmuje czas liniowy od liczby tych wielokrotności, czyli $\mathcal{O}(S)$, rozpropagowanie natomiast jednokrotnie zajmuje $O(n)$.

Problem L - Doliczony czas

Problem L - Doliczony czas

Zadanie

Mamy dane 5 sytuacji na boisku, z czego każda z nich zdarzyła się k_i razy, a za każde wystąpienie arbiter powinien doliczyć t_i sekund do meczu. Pytaniem jest ile minut musi ostatecznie doliczyć arbiter na koniec meczu.

Odpowiedź to

$$\left\lceil \frac{\sum_{i=1}^{5} k_i \cdot t_i}{60} \right\rceil.$$

Należało pamiętać o `long long`ach oraz poprawnym zaokrąglaniu w górę.

Problem M - Korzystna wymiana

Problem M - Korzystna wymiana

Zadanie

W zadaniu mamy dany zbiór liczb M, z których wszystkie są z przedziału $[0, 2^k - 1]$ i reprezentują maski bitowe. Następnie mamy dane dużo zapytań składających sie z dwóch liczb a i b, reprezentujących maski z tego samego przedziału. Zadanie polega na znalezieniu dla każdego takiego zapytania maski $m \in M$, która zarazem jest podmaską maski a (czyli każdy bit, który jest zapalony w masce m, musi też być zapalony w masce a), ale też m nie może być podmaską maski b.

Problem M - Korzystna wymiana

Zadanie

W zadaniu mamy dany zbiór liczb M, z których wszystkie są z przedziału $[0, 2^k - 1]$ i reprezentują maski bitowe. Następnie mamy dane dużo zapytań składających sie z dwóch liczb a i b, reprezentujących maski z tego samego przedziału. Zadanie polega na znalezieniu dla każdego takiego zapytania maski $m \in M$, która zarazem jest podmaską maski a (czyli każdy bit, który jest zapalony w masce m, musi też być zapalony w masce a), ale też m nie może być podmaską maski b.

Ograniczenia

$k = 20, Q, M \leqslant 10^6$

Problem M - Korzystna wymiana

Obserwacja

Jeśli znaleziona przez nas maska m nie może być podmaską maski b, to znaczy, że musi istnieć co najmniej jeden bit, który jest zapalony w masce m, za to nie jest zapalony w masce b.

Problem M - Korzystna wymiana

Obserwacja

Jeśli znaleziona przez nas maska m nie może być podmaską maski b, to znaczy, że musi istnieć co najmniej jeden bit, który jest zapalony w masce m, za to nie jest zapalony w masce b.

Zatem możemy dla każdego a oraz każdego bitu wyznaczyć podmaskę a, która nie ma teog bitu zapalonego.

Problem M - Korzystna wymiana

Programowanie dynamczne

Niech $DP[a][i]$ oznacza dowolną maskę z M, która jest podmaską a oraz ma zapalony bit i. Na początku możemy dla każdej maski $m \in M$ oraz dla każdego zapalonego w niej bitu i ustawić $DP[m][i] = m$. Następnie, obliczmy wartości DP w kolejności rosnącej liczby bitów a. Jeśli jakaś taka maska istnieje i nie jest nią samo a to musi ona być już uwzględniona w $DP[a']$ dla pewnego a' różniącej się od maski a jednym zgaszonym bitem. Możemy się po nich przeiterować.

Problem M - Korzystna wymiana

Złożoność

Przejścia zajmują $\mathcal{O}(k)$, liczba stanów to $k \cdot 2^k$. Odpowiadanie na zapytanie zajmuje $\mathcal{O}(k)$. Razem da to $\mathcal{O}(k^2 \cdot 2^k + Q \cdot k)$.

Usprawnienie

Za pomocą operacji bitowych możemy zarówno zmniejszyć czas przejścia, jak i zmniejszyć czas odpowiadania na zapytanie. Będzie to wtedy łącznie $\mathcal{O}(k \cdot 2^k + Q)$. Poprzednie rozwiązanie było jednak wystarczające.

Problem M - Korzystna wymiana

Alternatywne rozwiązanie

Klasyczne programowanie dynamiczne pozwala nam zliczyć ile jest podmasek a w zbiorze M dla wszystkich a w czasie $\mathcal{O}(k \cdot 2^k)$.

Problem M - Korzystna wymiana

Alternatywne rozwiązanie

Klasyczne programowanie dynamiczne pozwala nam zliczyć ile jest podmasek a w zbiorze M dla wszystkich a w czasie $\mathcal{O}(k \cdot 2^k)$.

Pytanie 1

Jak użyć tego do znajdowania rozwiązania zadania?

Problem M - Korzystna wymiana

Alternatywne rozwiązanie

Klasyczne programowanie dynamiczne pozwala nam zliczyć ile jest podmasek a w zbiorze M dla wszystkich a w czasie $\mathcal{O}(k \cdot 2^k)$.

Pytanie 1

Jak użyć tego do znajdowania rozwiązania zadania?

Pytanie 2

Jak zrobić zadanie w przypadku, gdyby zapytanie było postaci a, b, l i mielibyśmy wypisać l masek, które spełnia warunki? Interesuje nas złożoność $\mathcal{O}(k \cdot 2^k + k \cdot S)$, gdzie S jest sumą l po wszystkich zapytaniach

Problem N - Opaski

Problem N - Opaski

Zadanie

W zadaniu jest dane N par liczb, każda z zakresu od 0 do N. Liczba 0 oznacza, że dana pozycja jesst wolna. i-ta para jest poprawna, jeżeli na obu pozycjach jest i. Naszym zadaniem jest wypisać, ile można maksymalnie otrzymać poprawnych par przy optymalnych uzupełnieniu, w którym każda liczba występuje dwa razy.

Problem N - Opaski

Zadanie

W zadaniu jest dane N par liczb, każda z zakresu od 0 do N. Liczba 0 oznacza, że dana pozycja jesst wolna. i-ta para jest poprawna, jeżeli na obu pozycjach jest i. Naszym zadaniem jest wypisać, ile można maksymalnie otrzymać poprawnych par przy optymalnych uzupełnieniu, w którym każda liczba występuje dwa razy.

Obserwacja

Zauważmy, że wystarczy jedynie policzyć pary, które można uzupełnić poprawnie. Nie musimy się przejmować niepoprawnymi parami – je zawsze da się uzupełnić liczbami, których poprawne miejsce było już zajęte albo jedno z wystąpień było na złej pozycji.

Problem N - Opaski - rozwiązanie

Stwórzmy najpierw tablicę i zapiszmy do niej, jaka wartość znajduje się na każdej pozycji. Następnie obliczmy, ile razy występowała każda z liczb. Możemy to zrobić przy pomocy dodatkowej tablicy (gdzie i-ta pozycja to liczba wystąpień liczby i): przechodzimy po pierwszej tablicy i odpowiednio zwiększamy wartości w drugiej tablicy. Teraz wystarczy zliczyć pary, które można poprawnie uzupełnić. i-ta para może być poprawnie uzupełniona, gdy:

- obie pozycje są od początku poprawne (obie wartości są równe i),
- jedna pozycja to 0, druga to i oraz licznik wystąpień i jest równy jeden (jeżeli licznik byłby równy dwa, to druga liczba i jest już użyta gdzieś indziej),
- obie pozycje są równe 0 i licznik wystąpień i jest równy zero.

Problem O - Plan treningowy

Problem O - Plan treningowy

Zadanie

W zadaniu dane są trzy ciągi, każdy złożony z N zer i N jedynek, nazwijmy je A, B i C.
Niech pref_A oznacza ciąg sum prefiksowych stworzony z ciągu A.
Chcemy sprawić, żeby dla każdego i od 1 do $2 \cdot N$ zachodziło $\text{pref}_A[i] \leqslant \text{pref}_B[i] \leqslant \text{pref}_C[i]$. W tym celu możemy zamienić sąsiednie elementy w dowolnym z ciągów. Mamy wypisać, ile minimalnie trzeba wykonać takich operacji, żeby zachodziła powyższa zależność.

Problem O - Plan treningowy

Obserwacja

Niech $\text{pos}_A[i]$ oznacza pozycję i-tej jedynki w ciągu A. Zauważmy, że wymagana właściwość:

$$W_1 : \text{Dla każdego } i \text{ od } 1 \text{ do } 2 \cdot N : \text{pref}_A[i] \leqslant \text{pref}_B[i] \leqslant \text{pref}_C[i]$$

Zachodzi wtedy i tylko wtedy, gdy zachodzi:

$$W_2 : \text{Dla każdego } i \text{ od } 1 \text{ do } N : \text{pos}_A[i] \geqslant \text{pos}_B[i] \geqslant \text{pos}_C[i]$$

Problem O - Plan treningowy

Obserwacja

Ustalmy i, i popatrzmy na zamiany i-tych jedynek w każdym z trzech ciągów w jakimś optymalnym rozwiązaniu. Zauważmy, że jeżeli zamieniliśmy i-tą jedynkę w ciągu B w lewo, to zamiast tego możemy zamienić i-tą jedynkę w ciągu A w prawo, żeby zachować W_2. Analogicznie, zamiast zamiany w prawo w B, możemy zamienić w lewo w C. Powtarzając tą operację, otrzymamy rozwiązanie, w którym jedynki w ciągu B nie zmieniają swoich pozycji.

Problem O - Plan treningowy

Obserwacja

Ustalmy i, i popatrzmy na zamiany i-tych jedynek w każdym z trzech ciągów w jakimś optymalnym rozwiązaniu. Zauważmy, że jeżeli zamieniliśmy i-tą jedynkę w ciągu B w lewo, to zamiast tego możemy zamienić i-tą jedynkę w ciągu A w prawo, żeby zachować W_2. Analogicznie, zamiast zamiany w prawo w B, możemy zamienić w lewo w C. Powtarzając tą operację, otrzymamy rozwiązanie, w którym jedynki w ciągu B nie zmieniają swoich pozycji.

Rozwiązanie

Wystarczy zatem zsumować dla każdego i:

$$\max(0, pos_B[i] - pos_A[i]) + \max(0, pos_C[i] - pos_B[i])$$

Lista zadań