amppz

Omówienie zadań

AMPPZ 2025

amppz

H – Hakowanie

Najszybsze rozwiązanie:
UW1 (0:03)

Autor zadania:
Kamil Dębowski

H – Hakowanie

Zadanie

Od 9:00 do 19:00 jest 600 minut.

```
n = int(input())

a = [int(x) for x in input().split()]

print(600 - sum(a))
```

amppz

A – AIMPPZ

Najszybsze rozwiązanie:
Hiperkostki (UJ, 0:03)

Autorzy zadania:
Kamil Dębowski, Yahor Dubovik, Konrad Paluszek, Marcin Smulewicz, Jakub Onufry Wojtaszczyk

A – AIMPPZ

Treść

Wartość słowa to długość, razy 5 za każde „AI”.

$$f(\texttt{"AIXXXAI"}) = len \cdot 5^k = 7 \cdot 5^2 = 175$$

Znajdź najkrótsze słowo o wartości n.

A – AIMPPZ

Treść

Wartość słowa to długość, razy 5 za każde „AI”.

$$f(\texttt{"AIXXXAI"}) = len \cdot 5^k = 7 \cdot 5^2 = 175$$

Znajdź najkrótsze słowo o wartości n.

Rozwiązanie

```
AI = 0
while n % 5 == 0 and n // 5 >= 2 * (AI + 1):
        n //= 5
        AI += 1
```

amppz

I – Izolacja ICPC

Najszybsze rozwiązanie:
UWr7 (0:24)

Autor zadania:
Kamil Dębowski

I – Izolacja ICPC

Treść

Usadzić 80 drużyn (UW=11, UJ=9, UWR=8, ...) bez sąsiadowania. Zadana drużyna w lewym górnym rogu.

Kolizje na rysunku:

UWR	MAP	MAP	UJ	PL	UW	UW	SGGW	ZUT	GOO
MAP	PUT	UR	KUL	UW	UW	PM	AGH	UWR	AGH
PS	UJ	UWR	UW	UWR	MAP	UWR	PG	UWR	PWR
PW	HUA	PW	AGH	PG	DTP	UW	UW	UW	PWR
MAP	UG	NLU	UJ	PUT	UAM	UW	UJ	UJ	UW
NLU	ZUT	PW	WAT	UWR	AGH	UMCS	UR	UJ	AGH
PUT	UJ	SGGW	PG	PW	MAP	PO	MAP	UJ	UWR
UMCS	PO	PW	PW	UO	UW	PG	UJ	NLU	UMK

Trik

Możesz skopiować wyjście przykładu.

Trik

Możesz skopiować wyjście przykładu.

Rozwiązanie 1

Wylosuj ustawienie. Póki jest kolizja, zamień jedną kolidującą drużynę z losową.

Rozwiązanie 2

Podziel planszę na kwadraciki 2×2.

UWR		UWR		UWR		UWR		UWR	
UWR		UWR		UWR		UWR			

amppz

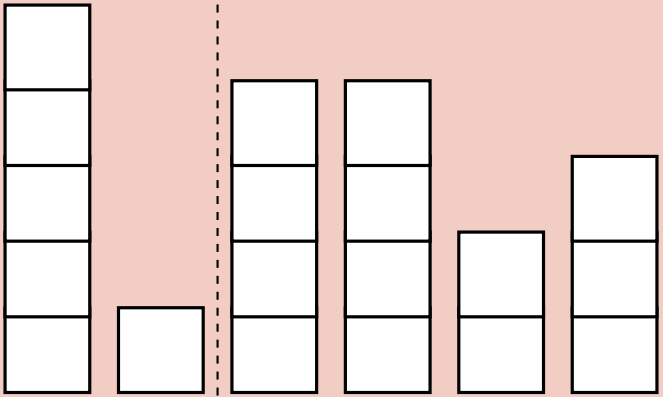
K – Klocki

Najszybsze rozwiązanie:
PW2 (0:06)

Autor zadania:
Jakub Onufry Wojtaszczyk

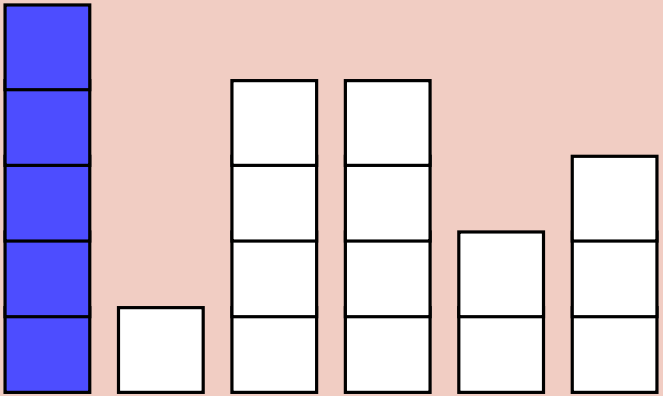
K – Klocki

Treść zadania

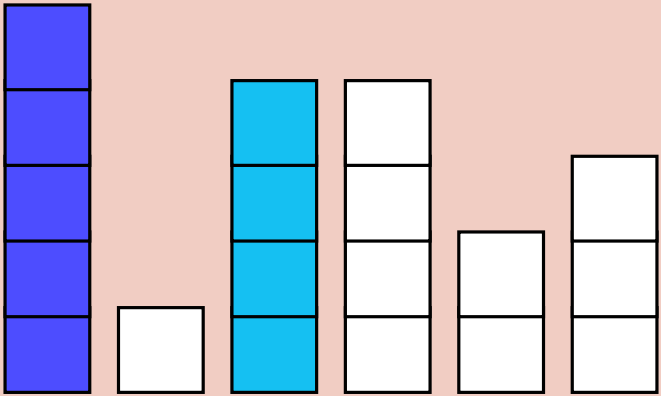


K – Klocki

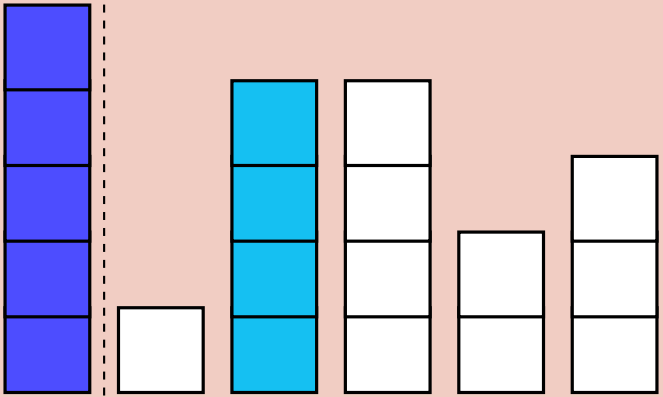
Najwyższa wieża



Druga najwyższa wieża



Podział



Wyszukiwanie

Można znaleźć dwa największe elementy na kilka sposobów:

- Jedna pętla, gdzie utrzymujemy indeksy dwóch największych dotychczas widzianych
- Prościej posortować i wziąć dwa pierwsze

Maksima prefiksów i sufiksów

amppz

J – Jury AMPPZ

Najszybsze rozwiązanie:
Jagiellonian 2 (0:56)

Autorzy zadania:
Yahor Dubovik, Jakub Onufry Wojtaszczyk

J – Jury AMPPZ

Treść zadania

- $N \leq 200\,000$ kandydatów zadań na AMPPZ, wybieramy K.
- Każde ma dotychczasową ocenę (w razie remisu kolejność)
- Dodajemy swoje oceny – *permutację* liczb $1 \ldots N$
- Chcemy wziąć jak najwięcej geometrii.

J – Jury AMPPZ

Treść zadania

- $N \leq 200\,000$ kandydatów zadań na AMPPZ, wybieramy K.
- Każde ma dotychczasową ocenę (w razie remisu kolejność)
- Dodajemy swoje oceny – *permutację* liczb $1 \ldots N$
- Chcemy wziąć jak najwięcej geometrii.

Pomysł

Wyszukiwanie binarne po wyniku.

Sortujemy geometrie po sumie.

Sortujemy niegeometrie po sumie.

J – Jury AMPPZ

Wyszukiwanie binarne

Czy możemy wziąc m geometrii?

- Chcemy m najlepszych.
- Ważne, czy najgorsza (po naszych ocenach) wejdzie.
- Wysokie oceny: $(n, n-1, \ldots)$.
- Sumy $(13, 15, 20) \to (13+n, 15+n-1, 20+n-2)$.

J – Jury AMPPZ

Wyszukiwanie binarne

Czy możemy wziąc m geometrii?

- Chcemy m najlepszych.
- Ważne, czy najgorsza (po naszych ocenach) wejdzie.
- Wysokie oceny: $(n, n-1, \ldots)$.
- Sumy $(13, 15, 20) \to (13+n, 15+n-1, 20+n-2)$.

Co wyrzucić?

Równoważnie — wyrzucamy *niegeo* $-(k-m)$ niegeometrii.

- Wyrzucamy te najgorsze.
- Niskie oceny $(1, 2, \ldots)$
- Sumy $(25, 22, 22) \to (25+1, 22+2, 22+3)$

amppz

F – Finanse

Najszybsze rozwiązanie:
UWr1 (0:47)

Autor zadania:
Yahor Dubovik

F – Finanse

Treść zadania

Dany spójny ważony nieskierowany graf $G(n, m)$

F – Finanse

Treść zadania

Dany spójny ważony nieskierowany graf $G(n, m)$ $(n, m \leq 10^6)$.

F – Finanse

Treść zadania

Dany spójny ważony nieskierowany graf $G(n, m)$ $(n, m \leq 10^6)$.
Bilanse pieniędzy w wierzchołkach $a_1, \ldots, a_n$

F – Finanse

Treść zadania

Dany spójny ważony nieskierowany graf $G(n, m)$ $(n, m \leq 10^6)$.
Bilanse pieniędzy w wierzchołkach $a_1, \ldots, a_n$ $(-10^9 \leq a_i \leq 10^9; \sum a_i = 0)$.

F – Finanse

Treść zadania

Dany spójny ważony nieskierowany graf $G(n, m)$ $(n, m \leq 10^6)$.
Bilanse pieniędzy w wierzchołkach $a_1, \ldots, a_n$ $(-10^9 \leq a_i \leq 10^9; \sum a_i = 0)$.
$A = \sum \max(a_i, 0)$

F – Finanse

Treść zadania

Dany spójny ważony nieskierowany graf $G(n, m)$ $(n, m \leq 10^6)$.
Bilanse pieniędzy w wierzchołkach $a_1, \ldots, a_n$ $(-10^9 \leq a_i \leq 10^9; \sum a_i = 0)$.
$A = \sum \max(a_i, 0)$
Krawędzie mają przepustowości $\frac{A}{2} \leq c_i$!

F – Finanse

Treść zadania

Dany spójny ważony nieskierowany graf $G(n, m)$ $(n, m \leq 10^6)$.
Bilanse pieniędzy w wierzchołkach $a_1, \ldots, a_n$ $(-10^9 \leq a_i \leq 10^9; \sum a_i = 0)$.
$A = \sum \max(a_i, 0)$
Krawędzie mają przepustowości $\frac{A}{2} \leq c_i$!
Czy da się doprowadzić bilanse do 0, przesyłając pieniądze krawędziami?

Przepływ

Przepływ

- Źródło

Przepływ

- Źródło + krawędzie do wierzchołków $a_i > 0$.

Przepływ

- Źródło + krawędzie do wierzchołków $a_i > 0$.
- Ujście

Przepływ

- Źródło + krawędzie do wierzchołków $a_i > 0$.
- Ujście + krawędzie do wierzchołków $a_i < 0$.

F – Finanse

Przepływ

- Źródło + krawędzie do wierzchołków $a_i > 0$.
- Ujście + krawędzie do wierzchołków $a_i < 0$.

Odpowiedź TAK $\Leftrightarrow$ Istnieje przepływ rozmiaru A.

F – Finanse

Przepływ

- Źródło + krawędzie do wierzchołków $a_i > 0$.
- Ujście + krawędzie do wierzchołków $a_i < 0$.

Odpowiedź TAK $\Leftrightarrow$ Istnieje przepływ rozmiaru A.

Max przepływ = Min przekrój

F – Finanse

Przepływ

- Źródło + krawędzie do wierzchołków $a_i > 0$.
- Ujście + krawędzie do wierzchołków $a_i < 0$.

Odpowiedź TAK $\Leftrightarrow$ Istnieje przepływ rozmiaru A.

Max przepływ = Min przekrój

Szukamy przekroju!

F – Finanse

Ile krawędzi z $c_i >= \frac{A}{2}$ w przekroju?

- 0 krawędzi

Ile krawędzi z $c_i >= \frac{A}{2}$ w przekroju?

- 0 krawędzi – Przekrój równy A.

F – Finanse

Ile krawędzi z $c_i >= \frac{A}{2}$ w przekroju?

- 0 krawędzi – Przekrój równy A.
- ≥ 2 krawędzi

Ile krawędzi z $c_i >= \frac{A}{2}$ w przekroju?

- 0 krawędzi – Przekrój równy A.
- ≥ 2 krawędzi – $c_i + c_j \geq \frac{A}{2} + \frac{A}{2} = A$

Ile krawędzi z $c_i >= \frac{A}{2}$ w przekroju?

- 0 krawędzi – Przekrój równy A.
- ≥ 2 krawędzi – $c_i + c_j \geq \frac{A}{2} + \frac{A}{2} = A$ – nie ma sensu.

F – Finanse

Ile krawędzi z $c_i >= \frac{A}{2}$ w przekroju?

- 0 krawędzi – Przekrój równy A.
- ≥ 2 krawędzi – $c_i + c_j \geq \frac{A}{2} + \frac{A}{2} = A$ – nie ma sensu.
- 1 krawędź

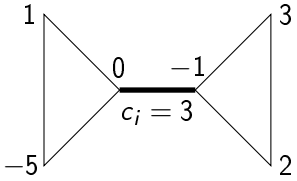
F – Finanse

Ile krawędzi z $c_i >= \frac{A}{2}$ w przekroju?

- 0 krawędzi – Przekrój równy A.
- ≥ 2 krawędzi – $c_i + c_j \geq \frac{A}{2} + \frac{A}{2} = A$ – nie ma sensu.
- 1 krawędź – Most w grafie.

Ile krawędzi z $c_i >= \frac{A}{2}$ w przekroju?

- 0 krawędzi – Przekrój równy A.
- ≥ 2 krawędzi – $c_i + c_j \geq \frac{A}{2} + \frac{A}{2} = A$ – nie ma sensu.
- 1 krawędź – Most w grafie.



F – Finanse

Algorytm

- szukamy mostów

Algorytm

- szukamy mostów
- bilanse poddrzew ($b_T = \sum_T a_i$)

Algorytm

- szukamy mostów
- bilanse poddrzew ($b_T = \sum_T a_i$)
- dla każdego mostu $|b_T| \leq c_i$?

F – Finanse

Algorytm

- szukamy mostów
- bilanse poddrzew ($b_T = \sum_T a_i$)
- dla każdego mostu $|b_T| \leq c_i$?

Złożoność

$$O(m)$$

amppz

B – Bity muzyczne

Najszybsze rozwiązanie:
MAP-2 (XIV LO Warszawa, 0:36), UW11 (0:58)

Autor zadania:
Marcin Smulewicz

B – Bity muzyczne

Treść zadania

Dana permutacja $a_1, a_2, \ldots, a_n$

B – Bity muzyczne

Treść zadania

Dana permutacja $a_1, a_2, \ldots, a_n$ – $(n \leq 200000)$

B – Bity muzyczne

Treść zadania

Dana permutacja $a_1, a_2, \ldots, a_n$ – $(n \leq 200000)$ Dozwolone operacje:

B – Bity muzyczne

Treść zadania

Dana permutacja $a_1, a_2, \ldots, a_n$ – $(n \leq 200000)$ Dozwolone operacje:

- Przenieś dowolny element na początek.

B – Bity muzyczne

Treść zadania

Dana permutacja $a_1, a_2, \ldots, a_n$ – $(n \leq 200000)$ Dozwolone operacje:

- Przenieś dowolny element na początek.
- Przenieś ostatni element w dowolne miejsce.

B – Bity muzyczne

Treść zadania

Dana permutacja $a_1, a_2, \ldots, a_n$ – $(n \leq 200000)$ Dozwolone operacje:

- Przenieś dowolny element na początek.
- Przenieś ostatni element w dowolne miejsce.

Posortuj ciąg

B – Bity muzyczne

Treść zadania

Dana permutacja $a_1, a_2, \ldots, a_n$ – $(n \leq 200000)$ Dozwolone operacje:

- Przenieś dowolny element na początek.
- Przenieś ostatni element w dowolne miejsce.

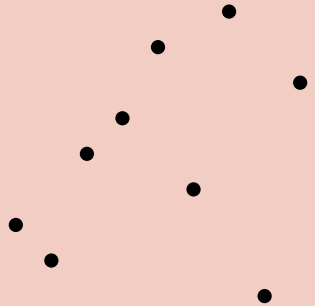
Posortuj ciąg – Minimalna liczba operacji.

B – Bity muzyczne

Przykład – $a = (3, 2, 5, 6, 8, 4, 9, 1, 7)$

B – Bity muzyczne

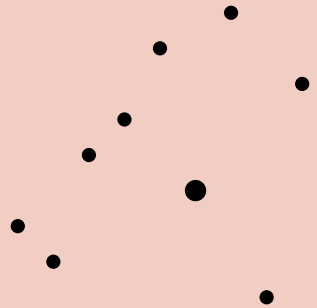
Przykład – $a = (3, 2, 5, 6, 8, 4, 9, 1, 7)$



Które elementy przeniesiemy ($\geq$ raz) ?

B – Bity muzyczne

Przykład – $a = (3, 2, 5, 6, 8, 4, 9, 1, 7)$

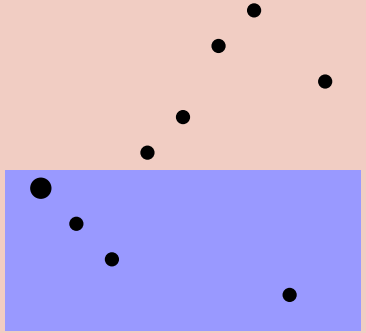


Które elementy przeniesiemy ($\geq$ raz) ?

- Największa wstawiona na początek
-

B – Bity muzyczne

Przykład – $a = (3, 2, 5, 6, 8, 4, 9, 1, 7)$

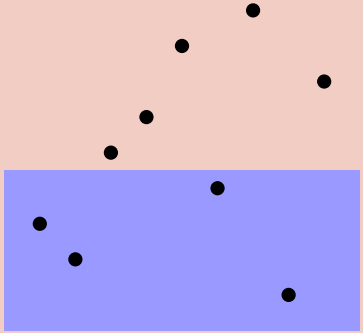


Które elementy przeniesiemy ($\geq$ raz) ?

- Największa wstawiona na początek + wszystkie mniejsze
-

B – Bity muzyczne

Przykład – $a = (3, 2, 5, 6, 8, 4, 9, 1, 7)$

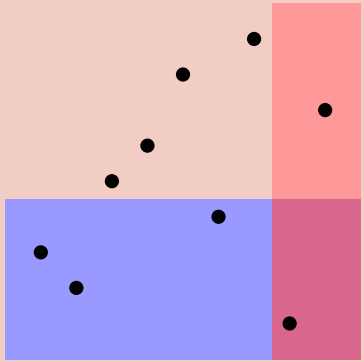


Które elementy przeniesiemy ($\geq$ raz) ?

- Największa wstawiona na początek + wszystkie mniejsze – prefiks wartości
-

B – Bity muzyczne

Przykład – $a = (3, 2, 5, 6, 8, 4, 9, 1, 7)$



Które elementy przeniesiemy ($\geq$ raz) ?

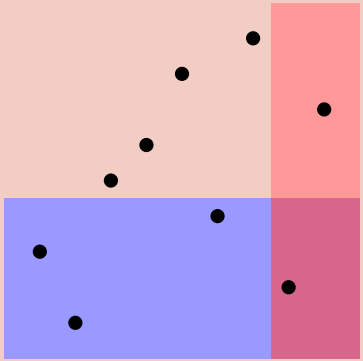
- Największa wstawiona na początek + wszystkie mniejsze – prefiks wartości
- Sufiks ciągu.

B – Bity muzyczne

Każdego przesuniemy raz

- prefiks wartość
- sufiks ciągu

Przykład

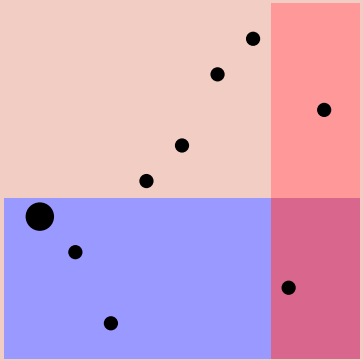


B – Bity muzyczne

Każdego przesuniemy raz

- prefiks wartość – na początek (od największych)
- sufiks ciągu

Przykład

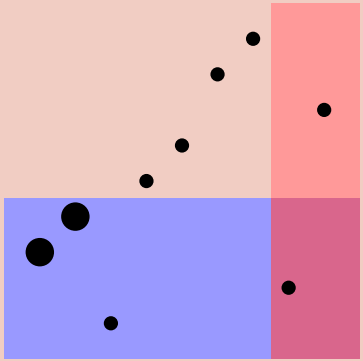


B – Bity muzyczne

Każdego przesuniemy raz

- prefiks wartość – na początek (od największych)
- sufiks ciągu

Przykład

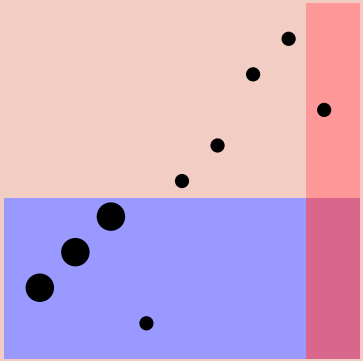


B – Bity muzyczne

Każdego przesuniemy raz

- prefiks wartość – na początek (od największych)
- sufiks ciągu

Przykład

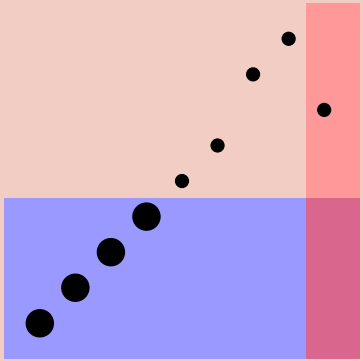


B – Bity muzyczne

Każdego przesuniemy raz

- prefiks wartość – na początek (od największych)
- sufiks ciągu

Przykład

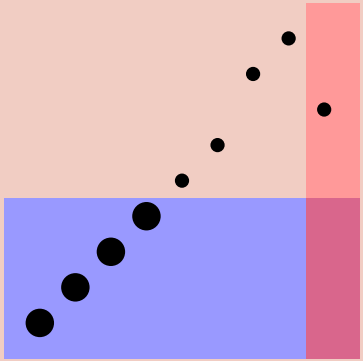


B – Bity muzyczne

Każdego przesuniemy raz

- prefiks wartość – na początek (od największych)
- sufiks ciągu – od końce (na swoje miejsce)

Przykład

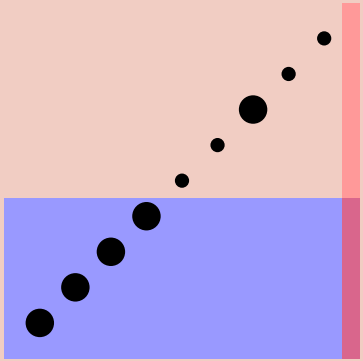


B – Bity muzyczne

Każdego przesuniemy raz

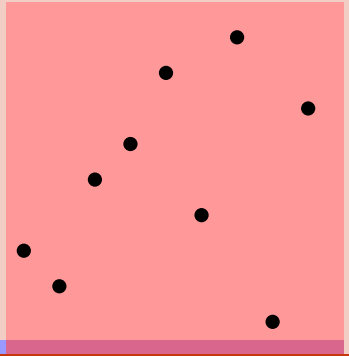
- prefiks wartość – na początek (od największych)
- sufiks ciągu – od końce (na swoje miejsce)

Przykład



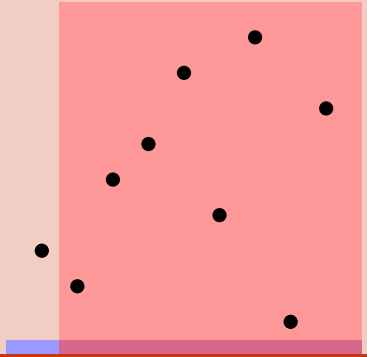
Algorytm

Iterujemy sufiks – utrzymujemy nieruszany ciąg.



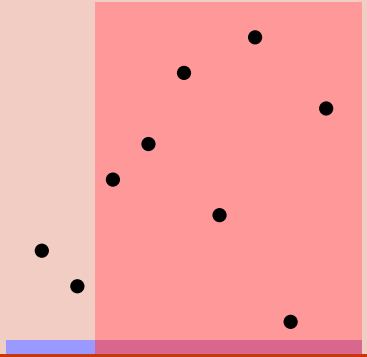
Algorytm

Iterujemy sufiks – utrzymujemy nieruszany ciąg.



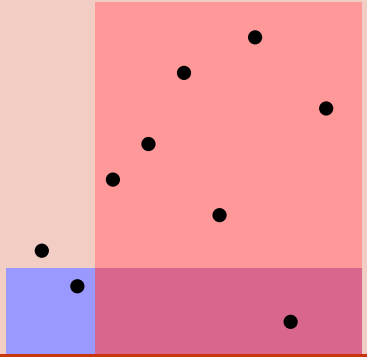
Algorytm

Iterujemy sufiks – utrzymujemy nieruszany ciąg.



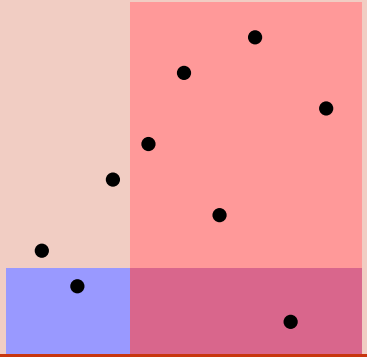
Algorytm

Iterujemy sufiks – utrzymujemy nieruszany ciąg.



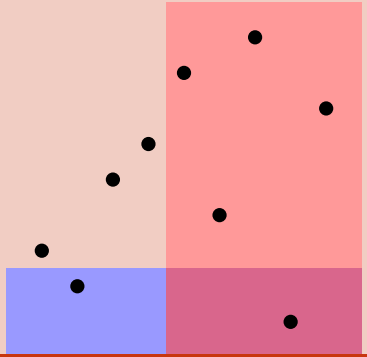
Algorytm

Iterujemy sufiks – utrzymujemy nieruszany ciąg.



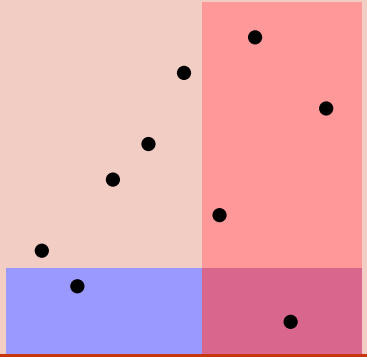
Algorytm

Iterujemy sufiks – utrzymujemy nieruszany ciąg.



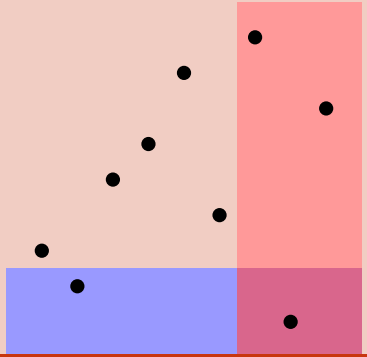
Algorytm

Iterujemy sufiks – utrzymujemy nieruszany ciąg.



Algorytm

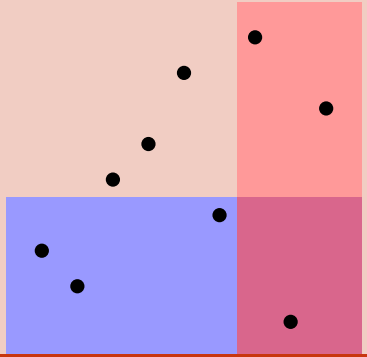
Iterujemy sufiks – utrzymujemy nieruszany ciąg.



B – Bity muzyczne

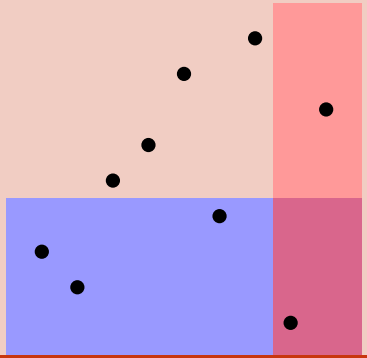
Algorytm

Iterujemy sufiks – utrzymujemy nieruszany ciąg.



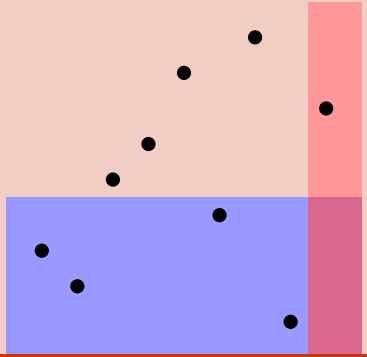
Algorytm

Iterujemy sufiks – utrzymujemy nieruszany ciąg.



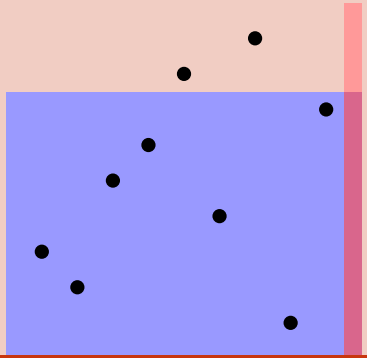
Algorytm

Iterujemy sufiks – utrzymujemy nieruszany ciąg.



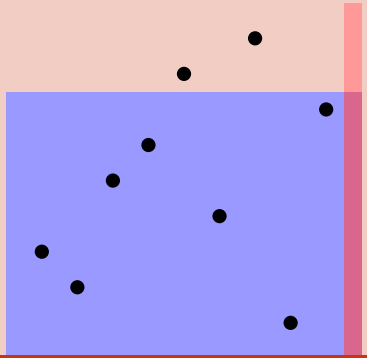
Algorytm

Iterujemy sufiks – utrzymujemy nieruszany ciąg.



Algorytm

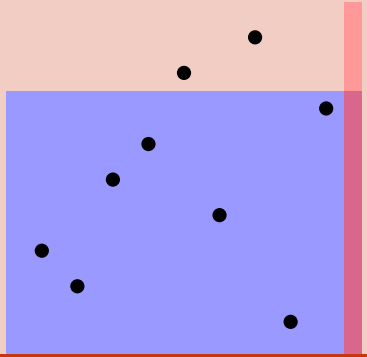
Iterujemy sufiks – utrzymujemy nieruszany ciąg.



Algorytm

Iterujemy sufiks – utrzymujemy nieruszany ciąg.
Nowy element:

- usuwamy z początku mniejsze
- dodajemy na koniec (jeśli jest największy)



Implementacja

- usuwamy z początku mniejsze
- dodajemy na koniec (jeśli jest największy)

Implementacja – Kolejka dwustronna

- usuwamy z początku mniejsze
- dodajemy na koniec (jeśli jest największy)

B – Bity muzyczne

Implementacja – Kolejka dwustronna

- usuwamy z początku mniejsze
- dodajemy na koniec (jeśli jest największy)

Złożoność

B – Bity muzyczne

Implementacja – Kolejka dwustronna

- usuwamy z początku mniejsze – $O(1)$ razy liczba usuwanych
- dodajemy na koniec (jeśli jest największy)

Złożoność

B – Bity muzyczne

Implementacja – Kolejka dwustronna

- usuwamy z początku mniejsze – $O(1)$ razy liczba usuwanych
- dodajemy na koniec (jeśli jest największy) – $O(1)$

Złożoność

B – Bity muzyczne

Implementacja – Kolejka dwustronna

- usuwamy z początku mniejsze – $O(1)$ razy liczba usuwanych
- dodajemy na koniec (jeśli jest największy) – $O(1)$

Złożoność

Dodajemy $O(n)$ razy $\Rightarrow$ Usuwany $O(n)$ razy.

Implementacja – Kolejka dwustronna

- usuwamy z początku mniejsze – $O(1)$ razy liczba usuwanych
- dodajemy na koniec (jeśli jest największy) – $O(1)$

Złożoność

Dodajemy $O(n)$ razy $\Rightarrow$ Usuwany $O(n)$ razy.

$$O(n)$$

amppz

E – Enigma

Najszybsze rozwiązanie:
UW10 (0:35)

Autor zadania:
Kamil Dębowski

E – Enigma

Treść zadania

Mamy sejf z 5-cyfrowym kodem na 5 tarczach. Odległością między dwoma kodami jest minimalna liczba przekręceń tarcz:

$$00000 \to 00010 \to 09010 \to 08010,$$

więc $dist(00000, 08010) = 3$.

E – Enigma

Treść zadania

Mamy sejf z 5-cyfrowym kodem na 5 tarczach. Odległością między dwoma kodami jest minimalna liczba przekręceń tarcz:

$$00000 \to 00010 \to 09010 \to 08010,$$

więc $dist(00000, 08010) = 3$.

Odległością kodu x od zbioru kodów S jest $dist(x, S) = \min_{y \in S} dist(x, y)$.

E – Enigma

Treść zadania

Mamy sejf z 5-cyfrowym kodem na 5 tarczach. Odległością między dwoma kodami jest minimalna liczba przekręceń tarcz:

$$00000 \to 00010 \to 09010 \to 08010,$$

więc $dist(00000, 08010) = 3$.

Odległością kodu x od zbioru kodów S jest $dist(x, S) = \min_{y \in S} dist(x, y)$.

Do zbioru S dorzucamy kolejno n kodów. Po każdym dorzuceniu wyznacz

$$\max_x dist(x, S)$$

i liczbę kodów x, które realizują tę odległość.

E – Enigma

Tworzymy nieskierowany graf o $V = 10^5$ wierzchołkach reprezentujących kody. Krawędzie są między kodami, które różnią się jednym przekręceniem tarczy.

E – Enigma

Tworzymy nieskierowany graf o $V = 10^5$ wierzchołkach reprezentujących kody. Krawędzie są między kodami, które różnią się jednym przekręceniem tarczy.

Dla $S = \{x\}$ robimy BFS-a z wierzchołka x i znajdujemy odległości do wszystkich pozostałych. Są one ograniczone przez $\frac{10}{2} \cdot 5 = 25$, więc suma odległości jest ograniczona przez $25 \cdot V$.

E – Enigma

Tworzymy nieskierowany graf o $V = 10^5$ wierzchołkach reprezentujących kody. Krawędzie są między kodami, które różnią się jednym przekręceniem tarczy.

Dla $S = \{x\}$ robimy BFS-a z wierzchołka x i znajdujemy odległości do wszystkich pozostałych. Są one ograniczone przez $\frac{10}{2} \cdot 5 = 25$, więc suma odległości jest ograniczona przez $25 \cdot V$.

Gdy dodajemy nowy x' do S, robimy nowego BFS-a z wierzchołka x', poprawiającego odległości (więc nie czyścimy tablicy odległości).

E – Enigma

Tworzymy nieskierowany graf o $V = 10^5$ wierzchołkach reprezentujących kody. Krawędzie są między kodami, które różnią się jednym przekręceniem tarczy.

Dla $S = \{x\}$ robimy BFS-a z wierzchołka x i znajdujemy odległości do wszystkich pozostałych. Są one ograniczone przez $\frac{10}{2} \cdot 5 = 25$, więc suma odległości jest ograniczona przez $25 \cdot V$.

Gdy dodajemy nowy x' do S, robimy nowego BFS-a z wierzchołka x', poprawiającego odległości (więc nie czyścimy tablicy odległości).

Każde dorzucenie wierzchołka do kolejki BFS wynika z poprawienia jakiejś odległości, a to możemy zrobić co najwyżej $25 \cdot V = 2\,500\,000$ razy.

E – Enigma

Tworzymy nieskierowany graf o $V = 10^5$ wierzchołkach reprezentujących kody. Krawędzie są między kodami, które różnią się jednym przekręceniem tarczy.

Dla $S = \{x\}$ robimy BFS-a z wierzchołka x i znajdujemy odległości do wszystkich pozostałych. Są one ograniczone przez $\frac{10}{2} \cdot 5 = 25$, więc suma odległości jest ograniczona przez $25 \cdot V$.

Gdy dodajemy nowy x' do S, robimy nowego BFS-a z wierzchołka x', poprawiającego odległości (więc nie czyścimy tablicy odległości).

Każde dorzucenie wierzchołka do kolejki BFS wynika z poprawienia jakiejś odległości, a to możemy zrobić co najwyżej $25 \cdot V = 2\,500\,000$ razy.

Trzymamy też tablicę $cnt[1..25]$, zawierającą liczności wierzchołków o danej odległości.

amppz

C – Ciągi trójkątne

Najszybsze rozwiązanie:
Hiperkostki (UJ, 1:25)

Autor zadania:
Yahor Dubovik

C – Ciągi trójkątne

Treść zadania

Ciągiem trójkątnym nazywamy każdy ciąg, który spełnia następujące warunki:

- Długość ciągu nie przekracza n.
- Każdy element jest liczbą całkowitą z przedziału $[1, n]$.
- Pewne trzy elementy tego ciągu są długościami boków jakiegoś niezdegenerowanego trójkąta.

Policz, ile istnieje ciągów trójkątnych.

C – Ciągi trójkątne

Jak sprawdzić, czy ciąg jest trójkątny?

Posortujemy ciąg.

$a_1 \leq a_2 \ldots \leq a_k$

Chcemy sprawdzać tylko trójki: a_i, a_{i+1}, a_{i+2}.

Ciąg będzie trójkątny, kiedy istnieje taki indeks $1 \leq i \leq k-2$:

$a_i + a_{i+1} > a_{i+2}$

C – Ciągi trójkątne

Nietrójkątne posortowane ciągi

$a_1 \leq a_2 \ldots \leq a_k$ (i)
$a_1 + a_2 \leq a_3$
$a_2 + a_3 \leq a_4$
...
$a_{k-2} + a_{k-1} \leq a_k$
Jak dodajemy warunek $a_1 \leq a_2$, to już wiemy, że ciąg będzie posortowany, nawet bez warunku (i).

Jak długi może być ciąg nietrójkątny?

$a_1 \geq 1$
$a_2 \geq a_1 \geq 1$
$a_3 \geq a_2 + a_1 \geq 1 + 1 = 2$
$a_4 \geq a_3 + a_2 \geq 2 + 1 = 3$
$a_5 \geq a_4 + a_3 \geq 3 + 2 = 5$
Liczby Fibonacciego:
$1, 1, 2, 3, 5, \ldots$
$Fib_{27} = 196418, Fib_{28} = 317811$
Długość ≤ 27.

Liczymy ciągi nietrójkątne

Nie jest łatwo policzyć ciągi a.
Wyrazimy a_i przez to, o ile jest większe od $a_{i-1} + a_{i-2}$.

$b_1 = a_1 > 0$
$b_2 = a_2 - a_1 \geq 0$
$b_3 = a_3 - a_2 - a_1 \geq 0$
$b_4 = a_4 - a_3 - a_2 \geq 0$
$\ldots$
$b_k = a_k - a_{k-1} - a_{k-2} \geq 0$

Liczymy ciągi nietrójkątne

Wyrazimy a_k przez $b_1, \ldots, b_k$

$a_k = b_k + a_{k-1} + a_{k-2}$

$a_k = b_k + 2b_{k-1} + 2a_{k-2} + a_{k-3}$

$a_k = b_k + 2b_{k-1} + 3b_{k-2} + a_{k-4}$

$\ldots$

$a_k = b_k + 2b_{k-1} + \ldots + Fib_k b_1$

$a = [1, 3, 10, 17, 45] \leftrightarrow b = [1, 2, 6, 4, 18]$

$a = [1, 3 + 1, 10 + 1, 17 + 2, 45 + 3] \leftrightarrow b = [1, 3, 6, 4, 18]$

Liczymy ciągi nietrójkątne

Chcemy policzyć liczbę ciągów $b_1, b_2, \ldots, b_k$
$b_1 + 2b_2 + \ldots + Fib_k b_k \leq N$
$O(kN^2)$ — łatwy plecak:
$DP[i][sum]$ — ile jest ciągów $b_1, \ldots, b_i$ takich, że $b_1 + \ldots + b_i Fib_i = sum$.
$O(kN)$ stanów, iterujemy się po wielkości b_i ($O(N)$).
Uważny czytelnik zauważy, że to działa w czasie $O(N^2)$, ale to wciąż za dużo.

Przyspieszamy dynamika

$Fib_i = t$

Przejścia:

$sum \to sum, sum + t, sum + 2t, \ldots$

$DP[i+1][sum + t] - DP[i+1][sum] = DP[i][sum + t]$

Działa w czasie $O(kN)$

C – Ciągi trójkątne

Liczymy ciągi trójkątne nieposortowane

Jeśli $a_1, \ldots, a_k$ są różne, to istnieje $k!$ różnych ciągów nieposortowanych.
$i \geq 3 \rightarrow a_i \geq a_{i-1} + a_{i-2} > a_{i-2}$
Jeśli $a_1 = a_2$, to istnieje $\frac{k!}{2}$ różnych ciągów.
$a_1 = a_2 \leftrightarrow b_{k-1} = 0$

C – Ciągi trójkątne

Cały algorytm

CiagiTrojkatne = Ciagi - CiagiNieTrojkatne
Ciagi = $N + N^2 \ldots + N^N$

CiagiNieTrojkatne

Iterujemy się po długości ciągu ($\leq 27 = O(\log(N))$):
Liczymy dynamik w czasie $O(N \log(N))$ (różne przejścia dla $b_{k-1} = 0$ i $b_{k-1} \neq 0$).
Czas $O(N \log^2(N))$
Da się policzyć dynamik tylko jeden raz, ale to nie było wymagane.

amppz

L – Liniowe uśrednianie

Najszybsze rozwiązanie:
UWr 2 (2:17)

Autor zadania:
Kamil Dębowski

L – Liniowe uśrednianie

Treść zadania

Dany ciąg (a_i) oraz liczba k.

Wybieramy przedział długości k i liniowo zmieniamy każdy element na średnią przedziału.

$$(1, 8) \to (4.5, 4.5)$$

L – Liniowe uśrednianie

Treść zadania

Dany ciąg (a_i) oraz liczba k.

Wybieramy przedział długości k i liniowo zmieniamy każdy element na średnią przedziału.

$$(1, 8) \to (4.5, 4.5)$$

Dla każdej liczby od 1 do m, jak szybko można ją gdzieś uzyskać?

Wynik to liczba rzeczywista w $[0, 1]$, lub -1.

Pomysł

Spójrzmy na $a_i = 5$ i średnie przedziałów z a_i.

$5 \to 12.8$ szybsze niż $5 \to 8.123$

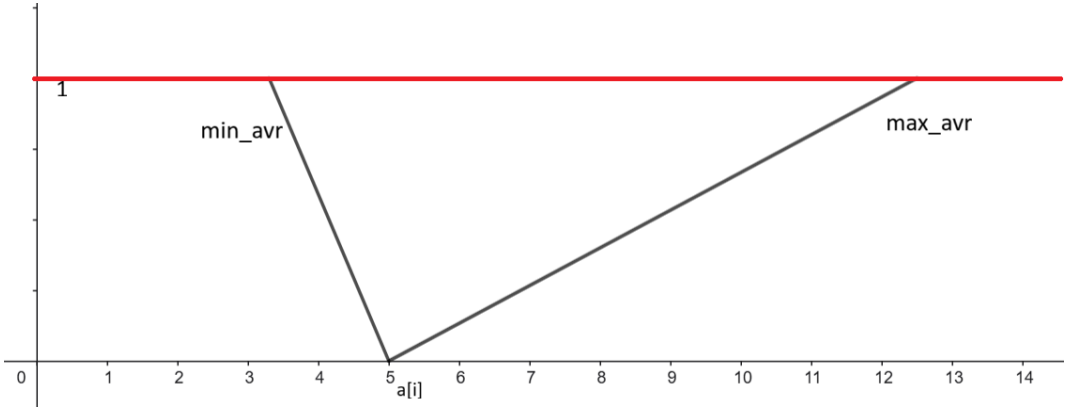
Pomysł

Spójrzmy na $a_i = 5$ i średnie przedziałów z a_i.

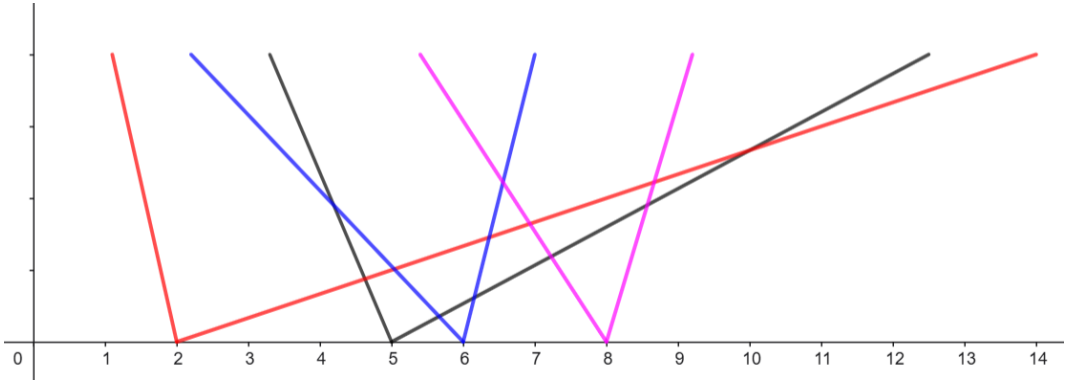
$5 \to 12.8$ szybsze niż $5 \to 8.123$

Dla każdego a_i, znajdź MIN i MAX średnią. Sumy prefiksowe + drzewo przedziałowe MIN i MAX.

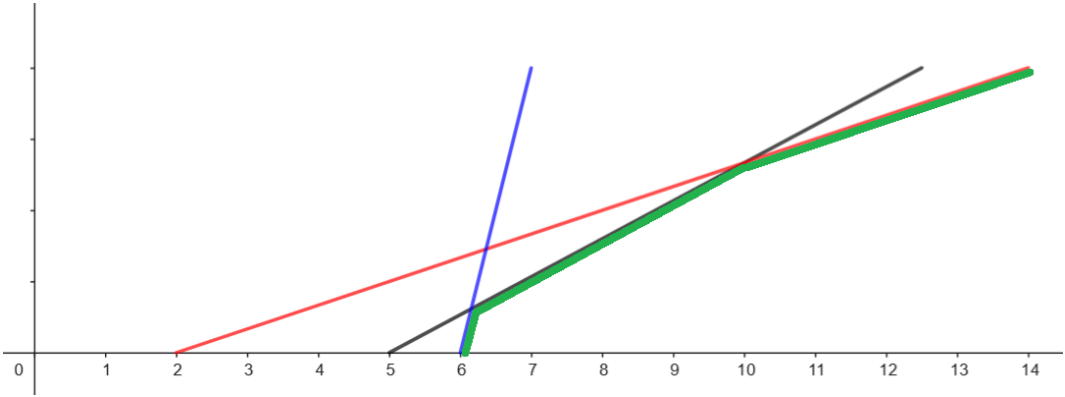
L – Liniowe uśrednianie



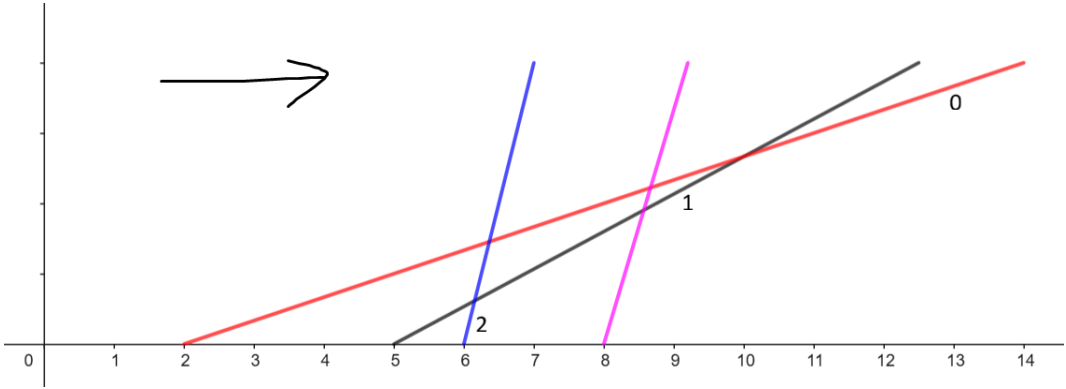
L – Liniowe uśrednianie



L – Liniowe uśrednianie



L – Liniowe uśrednianie



Złożoność

$$\mathcal{O}((n+q)\log n)$$

Pozwalaliśmy na $\log^2 n$ oraz $\sqrt{n}$.

Da się $\mathcal{O}(n+q)$, gdybyśmy umieli sortować liniowo.

amppz

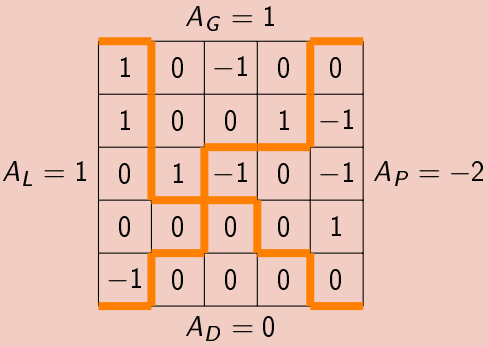
D – Dwie łamane

Najszybsze rozwiązanie:
UWr 1 (3:27)

Autor zadania:
Marek Sokołowski

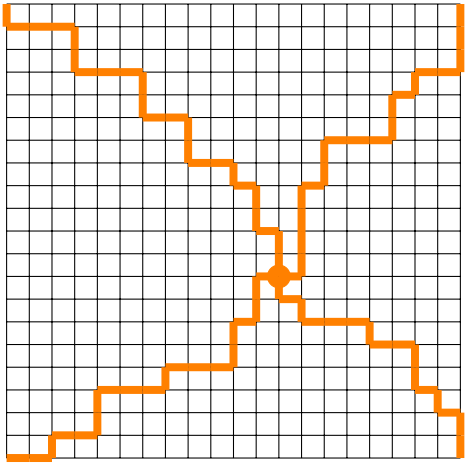
D – Dwie łamane

Treść zadania



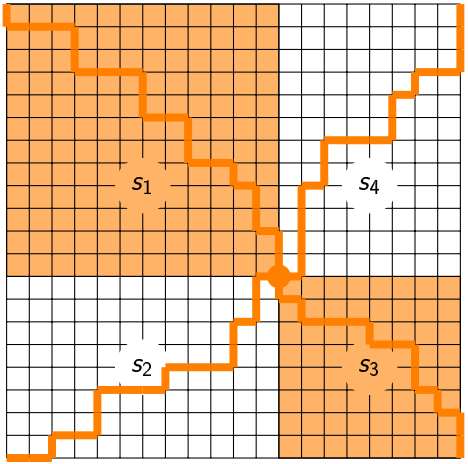
Podziel planszę $n \times n$ wypełnioną liczbami $-1, 0, 1$ na cztery obszary o ustalonych sumach dwoma łamanymi łączącymi przeciwległe wierzchołki.

D – Dwie łamane



Testujemy $(n+1)^2$ kandydatów na punkt przecięcia łamanych.

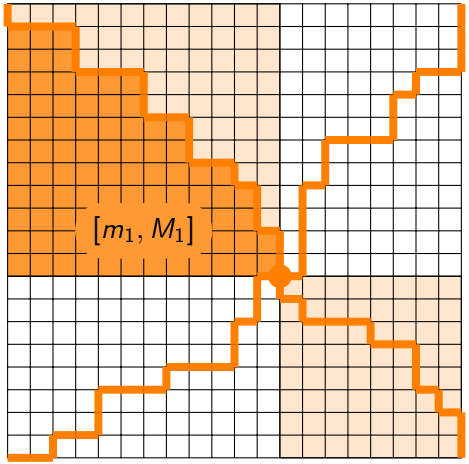
D – Dwie łamane



Testujemy $(n+1)^2$ kandydatów na punkt przecięcia łamanych.

Dzieli on planszę na cztery prostokąty o sumach s_1, s_2, s_3, s_4.

D – Dwie łamane

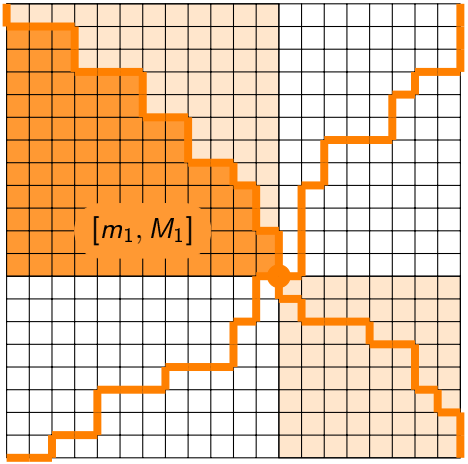


Testujemy $(n+1)^2$ kandydatów na punkt przecięcia łamanych.

Dzieli on planszę na cztery prostokąty o sumach s_1, s_2, s_3, s_4.

Rozważamy wszystkie fragmenty łamanej w lewym-górnym prostokącie i patrzymy na możliwe sumy w „trójkącie" pod łamaną. Z **ciągłości** wynika, że leżą one w pewnym przedziale $[m_1, M_1]$.

D – Dwie łamane



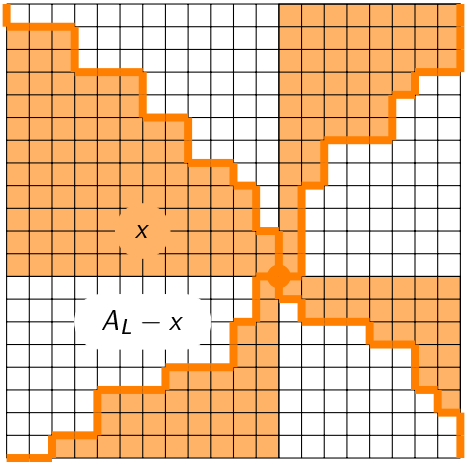
Testujemy $(n+1)^2$ kandydatów na punkt przecięcia łamanych.

Dzieli on planszę na cztery prostokąty o sumach s_1, s_2, s_3, s_4.

Rozważamy wszystkie fragmenty łamanej w lewym-górnym prostokącie i patrzymy na możliwe sumy w „trójkącie" pod łamaną. Z **ciągłości** wynika, że leżą one w pewnym przedziale $[m_1, M_1]$.

Analogicznie definiujemy $[m_i, M_i]$ dla $i = 2, 3, 4$.

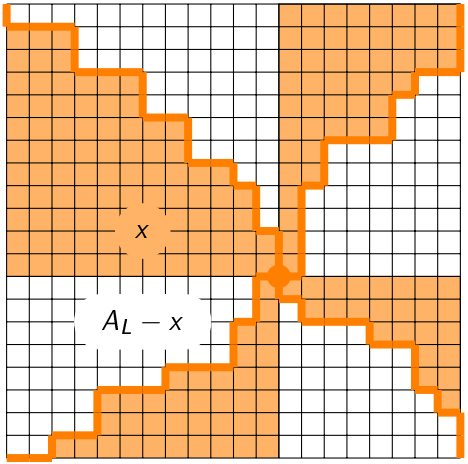
D – Dwie łamane



Jeśli ustalimy sumę $x \in [m_1, M_1]$ w tym trójkącie, to wymusza ona sumy w pozostałych siedmiu trójkątach, np. kolejne dwa mają sumy

$$A_L - x \quad \text{oraz} \quad s_2 - (A_L - x)$$

D – Dwie łamane



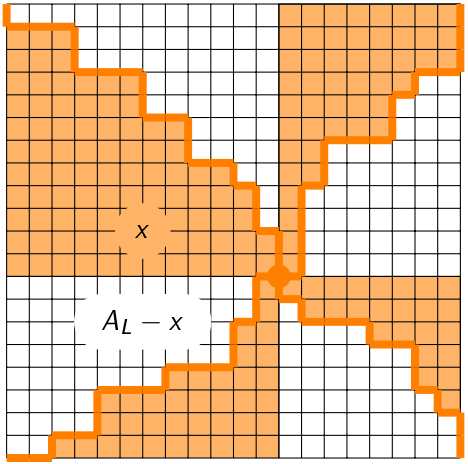
Jeśli ustalimy sumę $x \in [m_1, M_1]$ w tym trójkącie, to wymusza ona sumy w pozostałych siedmiu trójkątach, np. kolejne dwa mają sumy

$$A_L - x \quad \text{oraz} \quad s_2 - (A_L - x)$$

Zatem $s_2 - (A_L - x) \in [m_2, M_2]$, co daje drugi warunek

$$x \in [m_2 - s_2 + A_L, M_2 - s_2 + A_L].$$

D – Dwie łamane



Jeśli ustalimy sumę $x \in [m_1, M_1]$ w tym trójkącie, to wymusza ona sumy w pozostałych siedmiu trójkątach, np. kolejne dwa mają sumy

$$A_L - x \quad \text{oraz} \quad s_2 - (A_L - x)$$

Zatem $s_2 - (A_L - x) \in [m_2, M_2]$, co daje drugi warunek

$$x \in [m_2 - s_2 + A_L, M_2 - s_2 + A_L].$$

Jeśli cztery warunki na x są niesprzeczne, to odpowiedź jest pozytywna.

Złożoność czasowa

Programowaniem dynamicznym w czasie $O(n^2)$ wyznaczamy wartości s_1, m_1, M_1 dla wszystkich kandydatów na punkty przecięcia.

Złożoność czasowa

Programowaniem dynamicznym w czasie $O(n^2)$ wyznaczamy wartości s_1, m_1, M_1 dla wszystkich kandydatów na punkty przecięcia.

Obracamy planszę trzy razy i tak samo wyznaczamy s_i, m_i, M_i dla $i = 2, 3, 4$.

Złożoność czasowa

Programowaniem dynamicznym w czasie $O(n^2)$ wyznaczamy wartości s_1, m_1, M_1 dla wszystkich kandydatów na punkty przecięcia.

Obracamy planszę trzy razy i tak samo wyznaczamy s_i, m_i, M_i dla $i = 2, 3, 4$.

Mając te wartości, każdego kandydata sprawdzamy w czasie stałym.

Złożoność czasowa

Programowaniem dynamicznym w czasie $O(n^2)$ wyznaczamy wartości s_1, m_1, M_1 dla wszystkich kandydatów na punkty przecięcia.

Obracamy planszę trzy razy i tak samo wyznaczamy s_i, m_i, M_i dla $i = 2, 3, 4$.

Mając te wartości, każdego kandydata sprawdzamy w czasie stałym.

Ostatecznie złożoność to $O(n^2)$.

amppz

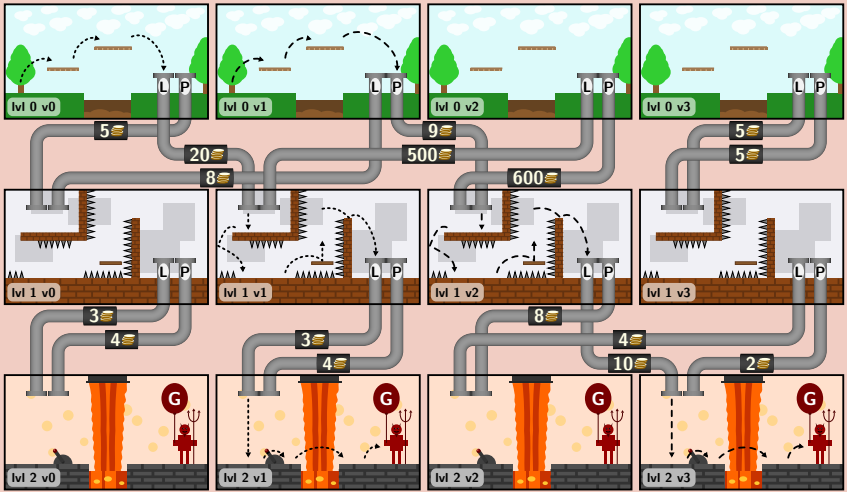
G – Gra w rury

Najszybsze rozwiązanie:
UWr 1 (4:08)

Autor zadania:
Bartosz Tarnawski

G – Gra w rury

Treść zadania



G – Gra w rury

Treść zadania

- Rozgrywkę opisuje początkowa wersja poziomu 0, oraz ciąg $\{L, P\}^n$.
- Wynikiem rozgrywki jest łączna zebrana liczba monet.
- Wykonujemy 300K rozgrywek testowych (i dostajemy wyniki).
- Potem dostajemy 10K rozgrywek, dla których mamy podać wyniki.

Chcielibyśmy odtworzyć cały graf z wejścia.

Wyniki nie wyznaczają grafu jednoznacznie

- Można dać $+1$ monetę na obu rurach wejściowych i -1 na obu rurach wyjściowych jakiejś wersji poziomu, i wyniki się nie zmienią.

G – Gra w rury

Chcielibyśmy odtworzyć cały graf z wejścia.

Wyniki nie wyznaczają grafu jednoznacznie

- Można dać $+1$ monetę na obu rurach wejściowych i -1 na obu rurach wyjściowych jakiejś wersji poziomu, i wyniki się nie zmienią.
- Mogą być dwie wersje jednego poziomu, które są nierozróżnialne (np. wszystkie wersje poziomu N).

G – Gra w rury

Chcielibyśmy odtworzyć cały graf z wejścia.

Wyniki nie wyznaczają grafu jednoznacznie

- Można dać $+1$ monetę na obu rurach wejściowych i -1 na obu rurach wyjściowych jakiejś wersji poziomu, i wyniki się nie zmienią.
- Mogą być dwie wersje jednego poziomu, które są nierozróżnialne (np. wszystkie wersje poziomu N).

Postać kanoniczna grafu

- Wszystkie rury `L` poza tymi z pierwszego poziomu mają zero monet.
- Utożsamiamy nierozróżnialne wierzchołki (będą miały więcej rur na wejściu).

Konstruujemy od końca

Ostatni poziom grafu znamy (jest jedna wersja).
Jak skonstruować poziom p znając $p+1$?

- Losujemy sporo początków długości p.
- Dla każdego chcemy się dowiedzieć:
  - Dokąd prowadzi rura `L`
  - Dokąd prowadzi rura `P`
  - Ile jest monet na rurze `P` (to jest proste)

Sufiks `LLL...` daje liczbę monet na dojściu do tego punktu.

Gdzie jestem na poziomie $p+1$?

Na początku mogę być w dowolnej wersji $p+1$. Muszę odróżnić, powiedzmy, wersje 0 i 1.

- Jeśli różnią się liczbą monet na prawej rurze: sufiks `PLLL`...
- Jeśli różnią się punktem docelowym rury `L`, to dodaję `L` i rozróżniam gdzie jestem na poziomie $p+2$.
- Podobnie jeśli różnią się punktem docelowym rury `P`.

G – Gra w rury

Liczba zapytań

- Dla każdego prefiksu, w $2 + 2(K - 1)$ zapytaniach poznam wszystkie parametry dokąd dochodzę.
- Stać mnie na $300\,000/2NK$ prefiksów na poziom, czyli około 375 prefiksów.
- Każdy prefiks trafia w dowolną wersję z równym prawdopodobieństwem.

Więc odtwarzamy cały graf od ostatniego poziomu w dół.

amppz

M – Mieszkanie

Najszybsze rozwiązanie:
UWr1 (4:56)

Autor zadania:
Kamil Dębowski

M – Mieszkanie

Treść zadania

- Cena mieszkania w dniu i to losowa liczba z $[a_i, b_i]$.
- Możemy mieć najwyżej jedno mieszkanie.
- Normalnie cenę poznajemy na początku dnia (i możemy kupić lub sprzedać)
- Przed pierwszym dniem możemy k cen
- Jaki jest maksymalny oczekiwany zysk?

Liczy się tylko następny dzień

Równoważnie, każdego dnia:

- Jeśli mam mieszkanie, to je sprzedaję, a potem mogę od razu odkupić.
- Jeśli nie mam mieszkania, to kupuję, a potem mogę od razu odsprzedać.

To oznacza, że decyzja w dniu i zależy tylko od cen w dniach i i $i+1$.

Zysk z decyzji w dniu i

W dniu i mam dwie możliwości:

- Kupuję mieszkanie w dniu i, żeby sprzedać je w dniu $i+1$
- Nie kupuję mieszkania w dniu i (i nie sprzedaję go w $i+1$).

W pierwszej opcji zysk to $c_{i+1} - c_i$, a w drugiej zero.

Muszę wybrać tak, żeby wartość oczekiwana była jak największa.

Jeśli nie znam ceny w dniu $i+1$

Do wyboru mam 0 oraz

$$\mathbb{E}(c_{i+1} - c_i) = \mathbb{E}(c_{i+1}) - c_i = \frac{a_{i+1} + b_{i+1}}{2} - c_i.$$

Zatem wartość oczekiwana zysku z dnia i to

$$\mathbb{E} \max \left\{ 0, \frac{a_{i+1} + b_{i+1}}{2} - c_i \right\}.$$

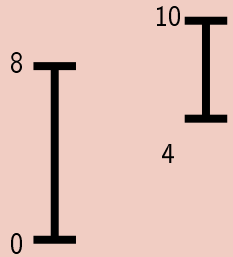
Jeśli znam cenę w dniu $i+1$

Wtedy wiem wszystko, i wybieram lepszą opcję, czyli mój zysk to $\max\{0, c_{i+1} - c_i\}$. Zatem wartość oczekiwana zysku z dnia i to

$$\mathbb{E} \max \{0, c_{i+1} - c_i\}.$$

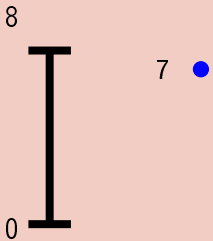
Jak to liczyć?

$$\mathbb{E} \max \left\{ 0, \frac{a_{i+1} + b_{i+1}}{2} - c_i \right\}.$$



Jak to liczyć?

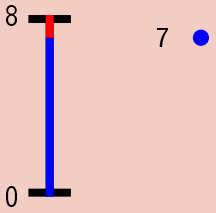
$$\mathbb{E} \max \left\{ 0, \frac{a_{i+1} + b_{i+1}}{2} - c_i \right\}.$$



Istotna jest tylko średnia odcinka $[a_{i+1}, b_{i+1}]$ równa $\frac{4+10}{2} = 7$.

Jak to liczyć?

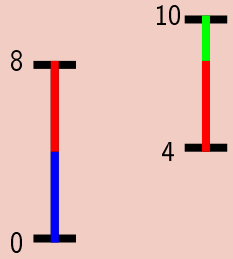
$$\mathbb{E} \max \left\{ 0, \frac{a_{i+1} + b_{i+1}}{2} - c_i \right\}.$$



Jest $\frac{7}{8}$ szansy, że trafimy w niebieski przedział (czyli poniżej 7), i wartość oczekiwana to $7 - \frac{0+7}{2} = \frac{7}{2}$. Jest też $\frac{1}{8}$ szansy, że trafimy w czerwony przedział, gdzie wartość oczekiwana to zero (bo nie kupimy mieszkania). Zatem wartość oczekiwana to $\frac{7}{8} \cdot \frac{7}{2} = \frac{49}{16}$.

Jak to liczyć?

$$\mathbb{E} \max \{0, c_{i+1} - c_i\} .$$



Podobnie (choć bardziej skomplikowanie) wyglądają wzory kiedy znamy cenę w dniu $i + 1$.

Jak zrobić zadanie?

Dla każdego dnia (od 1 do $N-1$) liczymy, jaki będzie zysk z tego dnia jeśli znamy dzień $i+1$ i jeśli nie znamy dnia $i+1$.
Sortujemy po różnicy między tymi dwoma liczbami, wybieramy wariant, w którym znamy dzień $i+1$ dla k największych różnic, i wypisujemy sumę.