Akademickie Mistrzostwa Polski w Programowaniu Zespołowym

Prezentacja rozwiązań zadań

30 października 2011

Czy się zatrzyma?

Autor zadania: Jakub Łącki

Zgłoszenia: 104 z 914 (11%)
Zaakceptowane przez 55 z 55 drużyn (100%)

Czy się zatrzyma?

$$
\begin{array}{l}
\textbf{while } n > 1 \textbf{ do} \\
\quad \textbf{if } n \bmod 2 = 0 \textbf{ then} \\
\quad\quad n := \lfloor n/2 \rfloor \\
\quad \textbf{else} \\
\quad\quad n := 3 \cdot n + 3
\end{array}
$$

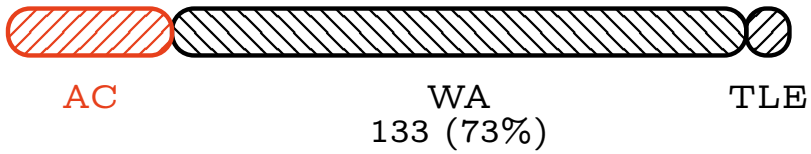
- Jeśli liczba jest podzielna przez 3, to już zawsze będzie.
- Jeśli liczba ma nieparzysty dzielnik, to w pewnym kroku stanie się podzielna przez 3.

Zatem TAK tylko dla potęg dwójki.
Złożoność czasowa $O(1)$.

Herbata z mlekiem

Autor zadania: Tomasz Idziaszek

Zgłoszenia: 181 z 914 (19%)
Zaakceptowane przez 38 z 55 drużyn (69%)



Herbata z mlekiem

Nie wykonujmy ostatniej dolewki (i tak jej nie wypijemy).
Niech h będzie liczbą dolewek herbaty, a m liczbą dolewek mleka.
Bajtazar wypije $(h+1)/2$ filiżanek herbaty minus to, co zostało na końcu (a zostało mniej niż $1/2$ filiżanki herbaty);
analogicznie dla mleka.

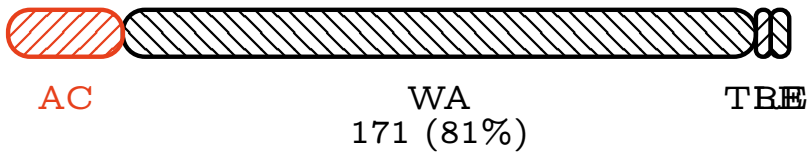
- Jeśli zatem $h > m$, to wypił więcej herbaty.
- Jeśli $h < m$, to wypił więcej mleka.
- Jeśli $h = m$, to patrzymy, czego zostało więcej na końcu.
  Z przedostatniej dolewki pochodzi $1/4$ filiżanki, więc jeśli $s_{n-1} = \mathtt{H}$, to wypił więcej mleka.
- Chyba że $n = 1$, wtedy wypił tyle samo mleka co herbaty.

Złożoność czasowa $O(n)$.

Krzyżak

Autor zadania: Szymon Acedański

Zgłoszenia: 211 z 914 (23%)
Zaakceptowane przez 31 z 55 drużyn (56%)



Krzyżak

Należy sprawdzić, czy pozycje much leżą w jednej płaszczyźnie.

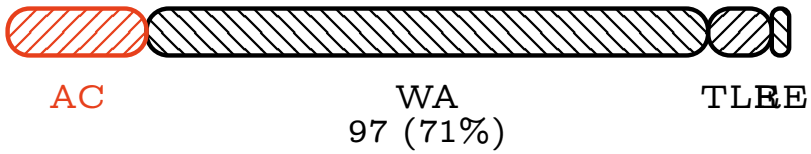
- Jeśli $n \leq 3$, to odpowiedzią jest TAK.
- Liczymy iloczyny wektorowe $p_1 - p_0 \times p_i - p_0$. Jeśli wszystkie są równe 0, to punkty leżą na jednej prostej – odpowiedź TAK.
- Bierzemy i dla niezerowego iloczynu i liczymy iloczyny mieszane $(p_1 - p_0 \times p_i - p_0) \cdot (p_j - p_0)$. Jeśli wszystkie są równe 0, to odpowiedź TAK, w przeciwnym wypadku NIE.

Złożoność czasowa $O(n)$.

Frania

Autor zadania: Szymon Acedański

Zgłoszenia: 135 z 914 (14%)
Zaakceptowane przez 24 z 55 drużyn (43%)



Frania

Osobie, która gromadziła pranie przez d dni, musimy przydzielić $5d$ klamerek jednego koloru albo $2d$ klamerek jednego i $3d$ innego koloru.
Rozwiązanie zachłanne:

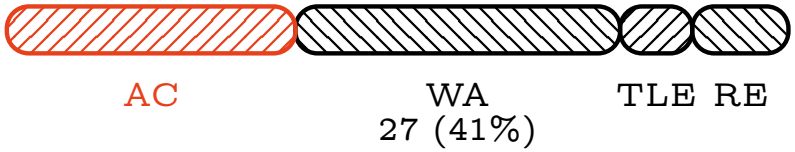
- Sortujemy osoby nierosnąco po d, a liczby klamerek trzymamy w zbiorze.
- Dla każdej kolejnej osoby o wymaganiu d, jeśli istnieje kolor zawierający co najmniej $5d$ klamerek, to używamy takiego koloru o najmniejszej możliwej liczbie klamerek, a w przeciwnym razie używamy dwóch kolorów zawierających odpowiednio co najmniej $3d$ i $2d$ klamerek, znów o najmniejszych możliwych liczbach klamerek.
- Jeśli w jakimś przypadku żądany kolor nie istnieje, zwracamy NIE.

Złożoność czasowa to $O(n \log n)$, a pamięciowa $O(n)$.

Eksterminacja świstaków

Autor zadania: Jakub Radoszewski

Zgłoszenia: 65 z 914 (7%)
Zaakceptowane przez 24 z 55 drużyn (43%)



Eksterminacja świstaków

Pozycje norek trzymamy posortowane w tablicy.
Pozycje magnetofonów trzymamy w zbiorze.
Pamiętamy aktualną liczbę norek a, które są w zasięgu muzyki.

- Jeśli dokładamy magnetofon na pozycji x, to liczymy, ile pokryje nowych norek. Znajdujemy w zbiorze pozycje x', x'' dwóch najbliższych magnetofonów ($x' < x < x''$). Wyszukiwaniem binarnym w tablicy znajdujemy liczbę norek w przedziale domkniętym

$$[\max(x' + l + 1, x - l), \min(x + l, x'' - l - 1)]$$

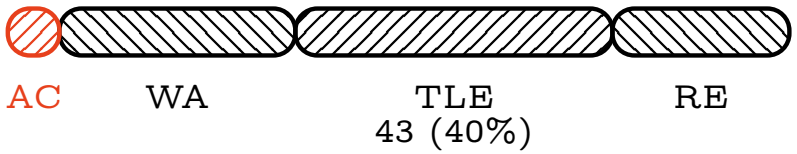
  Dodajemy tę wartość do a, a x do zbioru.
- Analogicznie przy usuwaniu magnetofonu.

Złożoność czasowa $O(n + m + d(\log n + \log m))$,
pamięciowa $O(n + m)$.

Jaskinia

Autor zadania: Jakub Łącki

Zgłoszenia: 106 z 914 (11%)
Zaakceptowane przez 7 z 55 drużyn (12%)



Jaskinia

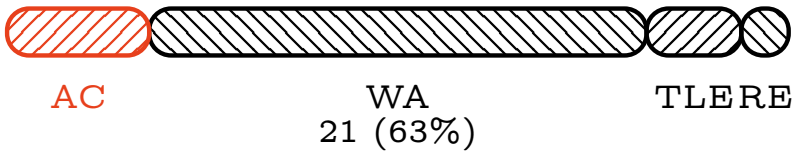
Drzewo można podzielić na spójne kawałki rozmiaru k wtedy i tylko wtedy, gdy istnieje w nim dokładnie $n/k - 1$ krawędzi łączących dwa poddrzewa o rozmiarach podzielnych przez k.

- Dla każdego $k \mid n$ przechodzimy drzewo i liczymy „dobre” krawędzie. Złożoność czasowa $O(n\varphi(n))$.
- Przechodzimy drzewo i dla każdej krawędzi (łączącej poddrzewa rozmiarów i oraz $n - i$) zwiększamy licznik dla każdego dzielnika liczby $\mathrm{nwd}(i, n - i)$. Robiąc to sprytnie, dostajemy złożoność $O(n \log n)$.
- Możemy zawczasu policzyć sobie wartości $\mathrm{nwd}(i, n - i)$ dla wszystkich i, używając algorytmu podobnego do sita Eratostenesa. Złożoność: $O(n \log \log n)$.

Iloraz inteligencji

Autor zadania: Marek Cygan

Zgłoszenia: 33 z 914 (3%)
Zaakceptowane przez 6 z 55 drużyn (10%)



Iloraz inteligencji

Tworzymy graf dwudzielny $G = (V_1, V_2, E)$, gdzie V_1 to studenci matematyki, V_2 to studenci informatyki, a krawędzie E łączą studentów, którzy się znają. Załóżmy na początek, że iloraz inteligencji każdego studenta jest równy 1. Wtedy w tym grafie musimy znaleźć największą klikę dwudzielną.

- Jest to równoważne znalezieniu największego zbioru niezależnego w dopełnieniu grafu G'.
- Największy zbiór niezależny w G' jest dopełnieniem najmniejszego pokrycia wierzchołkowego G', a liczność tego pokrycia jest równa liczności największego skojarzenia w G' (twierdzenie Königa).

Aby znaleźć największe skojarzenie tworzymy sieć przepływową z grafu G' przez nadanie każdej krawędzi przepustowości ∞ i dodanie źródła połączonego z wierzchołkami V_1 i ujścia połączonego z wierzchołkami V_2 krawędziami o jednostkowej przepustowości.

Iloraz inteligencji

W przypadku ogólnym:

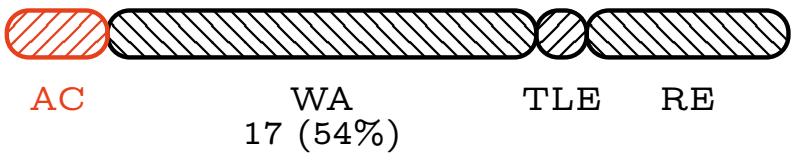
- Zastępujemy każdego studenta o ilorazie inteligencji q przez q identycznych kopii.
- Zauważamy, że w optymalnym rozwiązaniu dla każdej osoby albo bierzemy jej wszystkie kopie, albo żadną.
- Aby nie mieć gigantycznej sieci przepływowej, sklejamy z powrotem kopie danej osoby w jeden wierzchołek i łączymy go ze źródłem lub ujściem krawędzią o przepustowości q.

Złożoność czasowa (dla przepływu metodą Dinica) to $O((n+m)^3)$.

Prostokąt arytmetyczny

Autor zadania: Jakub Radoszewski

Zgłoszenia: 31 z 914 (3%)
Zaakceptowane przez 4 z 55 drużyn (7%)



Prostokąt arytmetyczny

Prostokąt o wysokości co najmniej 2 jest arytmetyczny wtedy i tylko wtedy, gdy wszystkie kolumny tego prostokąta są ciągami arytmetycznymi i pierwsze dwa wiersze tego prostokąta są ciągami arytmetycznymi.

Dla każdego pola (i, j) wyznaczamy długość najdłuższego ciągu arytmetycznego w dół – oznaczmy ją przez $t[i, j]$.

Teraz dla (i, j) będziemy chcieli sprawdzić, jaki jest największy prostokąt arytmetyczny o wysokości $t[i, j]$ zawierający (i, j) na górnym brzegu. Będzie on zawierał kolumny z przedziału $[a, b] \cap [\alpha, \beta]$, gdzie

- $[a, b]$ oznacza przedział, w którym w wierszach i oraz $i + 1$ są ciągi arytmetyczne,
- $[\alpha, \beta]$ oznacza przedział, w którym wartości $t[i, \cdot]$ są nie mniejsze niż $t[i, j]$.

Prostokąt arytmetyczny

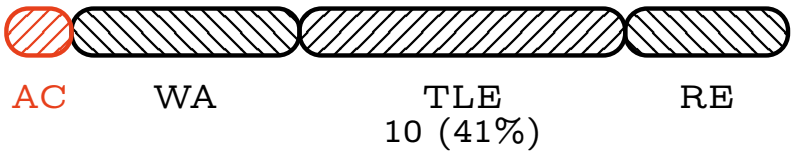
- Prostokąty arytmetyczne o wysokości 1, wartości $t[i, j]$ oraz przedziały $[a, b]$ obliczamy programowaniem dynamicznym w czasie $O(nm)$.
- Przedziały $[\alpha, \beta]$ dla każdego z wierszy liczymy w czasie $O(m)$ klasycznym sposobem znanym z zadania *Działka* z IX Olimpiady Informatycznej.

Złożoność czasowa i pamięciowa $O(nm)$.

Bajtocki Bieg Uliczny

Autor zadania: Jakub Łącki

Zgłoszenia: 24 z 914 (2%)
Zaakceptowane przez 2 z 55 drużyn (3%)



Bajtocki Bieg Uliczny

Dla każdego wierzchołka x definiujemy dwie ścieżki:

- ścieżkę dolną – idziemy z x, wybierając krawędzie w dół, a gdy nie jest to możliwe, to idziemy w prawo,
- ścieżkę prawą – idziemy, wybierając krawędzie w prawo, a gdy nie jest to możliwe, to idziemy w dół.

Ścieżki wyznaczamy implicite, dla każdego wierzchołka i dla każdego i zapamiętujemy, dokąd trafimy po wykonaniu 2^i kroków na ścieżce dolnej i prawej. Czas wstępnych obliczeń to $O(n \log n)$.

Bajtocki Bieg Uliczny

Chcemy sprawdzić, czy z wierzchołka p da się dojść do wierzchołka q, który leży na prawo i w dół od p.

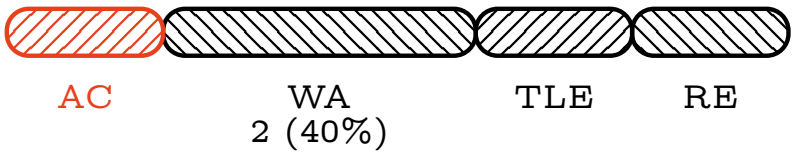
- Rozważamy prostokąt o rogach w p i q.
- Wyznaczamy punkt a, w którym ścieżka dolna z p dochodzi do brzegu prostokąta, oraz analogiczny punkt b dla ścieżki prawej z p. Robimy to w czasie $O(\log n)$.
- Odpowiedź jest TAK, wtedy i tylko wtedy gdy a leży na dolnej krawędzi prostokąta, a b na prawej krawędzi. (Bo wtedy ścieżka z 1 do q musi przecinać dolną lub prawą ścieżkę z p.)

Złożoność czasowa $O((n+k)\log n)$.

Drzewo i mrówki

Autor zadania: Szymon Acedański

Zgłoszenia: 5 z 914 (0%)
Zaakceptowane przez 1 z 55 drużyn (1%)



Drzewo i mrówki

- Wczytujemy rozmiar drzewa – znamy długość ścieżki.
- Wczytując wejście, symulujemy marsz Lewej Mrówki, do momentu pierwszego spotkania (dla każdej krawędzi sprawdzamy, czy doszło na niej do spotkania, bo wiedząc, jaką odległość w górę i w dół pokonała Lewa Mrówka, znamy dystans, jaki musiała pokonać Prawa).
- Wczytujemy wejście dalej, tym razem symulując marsz Prawej Mrówki do momentu, aż Lewa Mrówka wróci do korzenia (można pokazać, że zawsze będzie szybsza od Prawej).
- Dalej symulujemy marsz Prawej Mrówki aż do momentu drugiego spotkania.

Złożoność czasowa $O(n)$, pamięciowa $O(1)$.

Generator bitów

Autor zadania: Tomasz Idziaszek

Zgłoszenia: 19 z 914 (2%)
Zaakceptowane przez 0 z 55 drużyn (0%)

WA
16 (84%)

TLE

Generator bitów

- Tworzymy graf skierowany, w którym wierzchołkami są możliwe wartości ziarna i z każdego wierzchołka wychodzi dokładnie jedna krawędź do wierzchołka, który zawiera wartość uaktualnionego ziarna.
- Etykietujemy wierzchołki cyframi 0 i 1 (etykietą wierzchołka x jest 0, jeśli $x < \lfloor m/2 \rfloor$).
- Szukamy liczby takich wierzchołków x, że n-wierzchołkowa ścieżka zaczynająca się w następniku x ma listę etykiet zgodną z ciągiem $b_1, \ldots, b_n$.

Generator bitów

- Każdą słabo spójną składową grafu rozważamy osobno.
- Gdyby składowa była pojedynczą ścieżką, to algorytmem Knutha-Morrisa-Pratta szukalibyśmy ciągu $b_n, b_{n-1}, \ldots, b_1$ w słowie złożonym z etykiet odwróconej ścieżki w czasie $O(n+m)$.
- Gdyby składowa była drzewem, to moglibyśmy przeszukać drzewo w głąb, symulując algorytm na każdej ścieżce. Aby uzyskać złożoność $O(n+m)$ należy wcześniej spamiętać przejścia algorytmu KMP.
- W ogólnym przypadku (cykl z podczepionymi poddrzewami) rozwijamy cykl 3 razy i sprowadzamy zadanie do przypadku z drzewem, odpowiednio uwzględniając przypadek, gdy ciąg bitów nawija się wielokrotnie na cykl.

Złożoność czasowa i pamięciowa $O(n+m)$.

KONIEC