ARKAVIDIA

INFORMATICS & IT FESTIVAL

Editorial Penyisihan Competitive Programming Arkavidia 7.0

Contents

Problem A - Ambil Batu

Author: Kinantan Arya Bagaspati
Expected Difficulty: Medium

Bentuk soal ini cukup klasik karena mirip dengan NIM Game dengan solusi Dynamic Programming. Solusi tersebut ialah menyimpan array boolean yang pada index ke-i bernilai true jika orang pertama menang dengan jumlah batu i. Penghitungan DP tersebut juga cukup mudah, yakni

$$DP[i] = \neg(DP[i - A[0]] \vee DP[i - A[1]] \vee \cdots \vee DP[i - A[n-1]])$$

dengan array A ialah array yang diberikan pada soal.

Namun solusi ini tidak bisa digunakan karena tiap query ditanyakan jumlah batu hingga 10^{18}. Kuncinya ialah bilangan bilangan pada array A tidak ada yang lebih besar dari 20, sehingga penghitungan $DP[i]$ hanya bergantung pada 20 suku sebelumnya. Karena array DP berisi boolean, maka hanya terdapat 2^{20} kemungkinan subarray dengan panjang 20, sehingga jika kita menghitung $2^{20} + 20$ nilai pertama dari array DP ini, maka pasti terdapat 2 bilangan berbeda $x < y$ sehingga:

$$DP[x] = DP[y], DP[x+1] = DP[y+1], \cdots, DP[x+19] = DP[y+19]$$

Dengan begitu, kedua subarray ini pasti akan men-generate pola DP yang sama setelahnya. Solusi jadi hanya perlu mencari kedua subarray panjang 20 yang sama, selisihnya akan menjadi panjang cycle dimana pola berulang, sehingga tiap angka di query tinggal dimodulokan saja dengan panjang cycle apabila angka tersebut lebih besar dari x.

Kode Solusi: https://ideone.com/NOBpVE
Kompleksitas Waktu: $O(N \times 2^{A_{\max}})$
Catatan Panitia: Elon Musk dan Melvin Capital merupakan tokoh di drama "The GME Short Squeeze".

Problem B - Buku Raja

Author: Muhammad Hasan
Expected Difficulty: Medium

Kita dapat menyelesaikan soal ini dengan **Fenwick Tree/BIT**, anggap saja terdapat $M + N$ buah posisi kosong, awalnya kita taruh setiap buku di posisi $M + 1, M + 2, \ldots, M + N$ dan melakukan `add` untuk setiap posisi tersebut. Kemudian, untuk setiap query ke-i, kita cari buku nomor X dan kita taruh buku tersebut di posisi $M - i + 1$. Total waktu yang dibutuhkan bisa dicari dengan melakukan `query(posisi X)`, untuk lebih jelasnya dapat dilihat pada kode solusi.

Kode solusi: https://ideone.com/iCHw7I

Kompleksitas Waktu: $O(M \log(M + N))$

Catatan Panitia:

- Soal ini terinspirasi dari soal https://codeforces.com/contest/665/problem/B, yang kemudian dimodifikasi.
- Bisa digunakan `C++ STL PBDS` (Policy Based Data Structures) juga dalam menyelesaikan soal ini.
- Nama raja terinspirasi dari nama Perpustakaan Besar Alexandra di Mesir.

Problem C - Cokelat dan Permen

Author: Bimo Adityarahman Wiraputra
Expected Difficulty: Medium

Observasi yang dapat kita temukan adalah bahwa jika syarat dapat dipenuhi dengan X buah coklat dan permen untuk setiap anak, maka syarat juga dapat dipenuhi dengan $X+1$ buah cokelat dan permen untuk setiap anak.

Oleh karena itu, kita dapat melakukan *binary search* terhadap nilai X.

Jika kita memiliki nilai X, maka kita dapat memeriksa apakah syarat tersebut terpenuhi atau tidak. Pertama, urutkan pasangan (A_i, B_i) berdasarkan nilai B_i secara menurun. Bila nilai B_i sama, maka urutkan berdasarkan nilai A_i secara ascending. Lalu, iterasikan setiap pasangan dan buatlah array A selalu menaik secara *greedy*. Cara ini dijamin optimal karena nilai pada array B sudah bersifat non-ascending.

Kode Solusi: https://ideone.com/ZbS2Po

Kompleksitas Waktu: $O(N \log A_{\max} + B_{\max})$

Catatan Panitia:

Setelah rejudge, kita hanya menambah 1 buah kasus uji, yaitu kasus dimana semua nilai array A dan B sama.

Problem D - DVD

Author: Kinantan Arya Bagaspati
Expected Difficulty: Medium

Alih alih melihat sinar pada soal memantul-mantul pada dinding, cobalah memandang soal sebagai sinar yang lurus, serta cerminkanlah persegi panjang pada soal (sebutlah sebagai "box") di setiap sisinya infinitely-many (sehingga dapat divisualkan sebagai grid of boxes). Dari hasil cerminan terbentuk 4 jenis box, yakni box awal, horizontal-mirrored-box, vertical-mirrored-box, dan both-mirrored-box. Perhatikan bahwa dalam grid of boxes yang kita ciptakan, pola jenis box akan berulang tiap 2×2 box ($2N \times 2M$). Jika kemudian kita bicara dalam bahasa koordinat dimana entry matrix pada pojok kiri atas box awal ialah $(1, 1)$ dan pojok kanan bawah box awal ialah (N, M), maka entry pada koordinat (x, y) akan dicerminkan ke: horizontal-mirrored-box: $(x, 2M + 1 - y)$ vertical-mirrored-box: $(2N + 1 - x, y)$ both-mirrored-box: $(2N + 1 - x, 2M + 1 - y)$ Mengingat pola grid of boxes yang sudah ada, entry ini akan dicerminkan di semua koordinat yang berbentuk:

$$(i(2N) + (x \vee (2N + 1 - x)), j(2M) + (y \vee (2M + 1 - y)))$$

untuk setiap bilangan bulat i dan j

Jelas bahwa sinar akan melewati semua koordinat dengan bentuk (a, a), dan akan berhenti (mengenai sudut) bila a habis dibagi N dan M, yakni saat $a = lcm(N, M)$. Mudah dibuktikan pula bahwa setiap entry pada matrix dilewati sinar hanya maksimal sekali saja.

Bukti. Kita kembali ke sudut pandang sinar memantul-mantul. Sinar dengan arah kiri-bawah dan kanan-atas hanya menyentuh koordinat (x, y) dengan $x + y$ ganjil, sementara sinar dengan arah kiri-atas dan kanan-bawah hanya menyentuh koordinat (x, y) dengan $x + y$ genap. Setelah itu cukup jelas bahwa apabila terdapat entry yang terkena sinar 2 kali, jelas bahwa sinar tersebut telah menyentuh sudut.

Jelas pula bahwa bilangan yang tertera pada koordinat (a, a) ialah:

$$\left(\left\lfloor \frac{a-1}{N} \right\rfloor + \left\lfloor \frac{a-1}{M} \right\rfloor\right) \%9 + 1$$

Maka soal telah di reduce menjadi mencari bilangan bulat i dan j sehingga fakta berikut berlaku:

$$i(2N) + (x \vee (2N + 1 - x)) = j(2M) + (y \vee (2M + 1 - y)) \leq lcm(n, m)$$

Untuk itu diperlukan fungsi bernama **bezout** (sebuah teorema mengenai metode Euclid pada 2 bilangan) yang apabila diinputkan bilangan $2N$ dan $2M$ akan mengembalikan p dan q sehingga $p(2N) + q(2M) = gcd(2N, 2M)$ Setelah itu akan dicoba 4 kasus, dimana masing-masing kasusnya mengganti value U menjadi x atau $2N + 1 - x$ dan V menjadi y atau $2M + 1 - y$. Pada tiap kasus akan dicari nilai i dan j sehingga $i(2N) + (-j)(2M) = V - U$ dengan bantuan bezout.

Akhir kata, semua informasi di atas telah cukup untuk menyelesaikan soal.

Kode Solusi: https://ideone.com/QAERrw

Kompleksitas Waktu: $O(\log(\max(N, M)))$

Catatan Panitia: Buat yang belum sadar, pola yang dibentuk sinar merupakan pola DVD player screensaver.

Problem E - Emordnilap

Author: Chintya Wijaya
Expected Difficulty: Easy

Soal ini dapat diselesaikan dengan mengiterasikan string input. Jika menemukan karakter selain alfanumerik, keluarkan string "Emor tidak beruntung :(". Jika menemukan huruf kapital, ubah huruf tersebut menjadi huruf kecil. Lalu, keluarkan string tersebut secara terbalik.

Kode Solusi: https://ideone.com/p5ZzYz
Kompleksitas Waktu: $O(|S|)$
Catatan Panitia: FYI, Emordnilap merupakan kebalikan dari Palindrome.

Problem F - Farming Buah

Author: Muhammad Kamal Shafi
Expected Difficulty: Medium

Soal ini dapat diselesaikan dengan observasi sederhana. Perhatikan bahwa untuk memanen buah dengan jarak yang minimum, terdapat edge yang dilewati 1 kali dan terdapat edge yang dilewati 2 kali. Edge yang dilewati 1 kali akan membentuk path dari sebuah leaf ke leaf lainnya. Maka dari itu, diperlukan path dari leaf ke leaf yang paling jauh, sehingga jarak yang ditempuh adalah jarak minimum.

Sehingga jarak tempuh minimum (D_{min}):

$$D_{min} = 2 \sum_{i=1}^{N-1} W_i - \sum_{i \in \text{Diameter}} W_i$$

Setelah itu, untuk membentuk path keseluruhan yang minimum secara leksikografis dapat dilakukan dengan greedy. Yaitu, dengan memberi tanda pada node yang dapat menjadi bagian dari diameter.

Kode solusi: https://ideone.com/KVeGYm
Kompleksitas Waktu: $O(N \log N)$.
Catatan Panitia: Jeremiah Gottwald

sumber: https://www.pinterest.com/pin/403424079112028538/

Problem G - GCD Mania

Author: Muhammad Kamal Shafi
Expected Difficulty: Easy-Medium

Untuk menyelesaikan soal ini, akan dilakukan observasi terlebih dahulu.
Misal terdapat 2 bilangan positif A dan B $(A \leq B)$, maka akan berlaku persamaan berikut,

$$GCD(A, B) = GCD(A, B - A)$$

Hal tersebut juga tetap berlaku apa bila terdapat tiga bilangan A, B dan C $(A \leq B \leq C)$,

$$GCD(A, B, C) = GCD(A, B - A, C - A)$$

Dari situ terlihat bahwa untuk N buah bilangan terurut tidak menurun akan berlaku,

$$GCD(A_1, \ldots, A_i, \ldots, A_N) = GCD(A_1, \ldots, A_i - A_1, \ldots, A_N - A_1)$$

Sehingga apabila seluruh bilangan ditambahkan dengan sebuah bilangan X akan berlaku:

$$GCD(A_1 + X, \ldots, A_i + X, \ldots, A_N + X) = GCD(A_1 + X, \ldots, A_i - A_1, \ldots, A_N - A_1)$$
$$GCD(A_1 + X, \ldots, A_i + X, \ldots, A_N + X) = GCD(A_1 + X, GCD(A_2 - A_1, \ldots, A_i - A_1, \ldots, A_N - A_1))$$

Maka untuk mendapatkan hasil GCD maksimum dalam setiap query, dapat dilakukan iterasi terhadap faktor-faktor $GCD(A_2 - A_1, \ldots, A_i - A_1, \ldots, A_N - A_1)$ dan kemudian mengecek apakah terdapat X $(L \leq X \leq R)$ yang dapat menghasilkan GCD sebesar faktor tersebut.

Kode solusi: https://ideone.com/ICFcn8
Kompleksitas Waktu: $O(Q \log A)$.
Catatan Panitia: Gaia adalah singkatan dari GCD Mania.

Problem H - Hadiah untuk Kembaran

Author: Muhammad Hasan
Expected Difficulty: Easy

Soal ini cukup mudah, kita hanya perlu mengetahui KPK dari nomor favorit budi dan dono untuk mengetahui nomor mana saja yang tidak akan diambil. Menghitung KPK dari dua angka X dan Y, dapat dihitung dengan GCD/FPB sebagai berikut:

$$KPK(X,Y) = \frac{X \times Y}{FPB(X,Y)}$$

Kita akan dapat bahwa jawaban dari soal ini adalah:
Banyaknya hadiah yang diambil Budi (H_{Budi}):

$$\begin{aligned} H_{Budi} &= \text{Banyaknya hadiah yang habis dibagi X} - \text{Banyaknya hadiah yang habis dibagi X dan Y} \\ &= \left\lfloor \frac{N}{X} \right\rfloor - \left\lfloor \frac{N}{KPK(X,Y)} \right\rfloor \end{aligned}$$

Banyaknya hadiah yang diambil Dono (H_{Dono}):

$$\begin{aligned} H_{Dono} &= \text{Banyaknya hadiah yang habis dibagi Y} - \text{Banyaknya hadiah yang habis dibagi X dan Y} \\ &= \left\lfloor \frac{N}{Y} \right\rfloor - \left\lfloor \frac{N}{KPK(X,Y)} \right\rfloor \end{aligned}$$

Untuk implementasinya, kita perlu menggunakan `integer 64-bit`, dalam menghitung KPK.

Kode solusi: https://ideone.com/mbQNab
Kompleksitas Waktu: $O(\log N)$ untuk setiap testcase.
Catatan Panitia: Pembuat soal juga kembar.

Problem I - Indra dan Pizza

Author: Ryo Richardo
Expected Difficulty: Easy-Medium

Solusi dari permasalahan ini dapat diselesaikan menggunakan **linked-list**. Pada solusi ini, mula-mula definisikan tipe bentukan linked list yang mengandung `array el` untuk menyimpan `id` setiap piring pizza, `count` untuk menyimpan jumlah piring dalam tumpukan, `id` untuk menyimpan urutan tumpukan dari tumpukan pertama, dan `next` untuk menyimpan alamat tumpukan selanjutnya.

Kemudian buatlah fungsi dan prosedur yang dapat mengkoordinasikan piring-piring pizza ke dalam tumpukan sesuai permintaan, yaitu fungsi `Alokasi` (membentuk tumpukan baru), fungsi `Search` (mencari tumpukan dengan `id` tertentu), prosedur `Insert` (memanggil fungsi `Alokasi`, kemudian memasukan tumpukan yang terbentuk sebagai elemen baru linked-list), dan prosedur `Enqueue` (memasukan piring dengan `id` tertentu ke tumpukan dengan `id` tertentu; jika `id` tumpukan tidak ditemukan, maka memanggil prosedur `Insert`)

Pada program utama, mula-mula masukkan nilai N, M, dan Q sesuai spesifikasi soal. Kemudian, untuk menerima kondisi tumpukan awal, digunakan prosedur `Insert` (tumpukan ke linked-list) dan `Enqueue` (piring ke tumpukan). Kemudian, untuk query jenis 1, gunakan `Enqueue` untuk memasukkan piring ke tumpukan. Untuk query jenis 2, gunakan fungsi `Search` untuk mencari tumpukan dengan index tertentu, yang kemudian dikurangi elemen teratasnya. Untuk query jenis 3, gunakan fungsi `Search` untuk mencari kedua `id` tumpukan, serta prosedur `Enqueue` yang diiterasikan dari tumpukan satu ke tumpukan yang lainnya.

Setelah selesai menjalankan semua query, program menampilkan tumpukan yang memiliki minimal satu buah piring. Caranya adalah dengan mengiterasikan semua elemen pada linked-list, melakukan filter untuk tumpukan dengan 0 piring, kemudian melakukan iterasi untuk setiap piring yang ada pada tumpukan tersebut.

Kita juga dapat menggunakan STL List pada C++ yang dapat dilihat di dokumentasi C++. Gunakan `push_back` untuk query jenis 1, `pop_back` untuk query jenis 2, dan splice untuk query jenis 3.

Kode Solusi: https://ideone.com/eyWE8o
Kompleksitas Waktu: $O(N)$
Catatan Panitia:

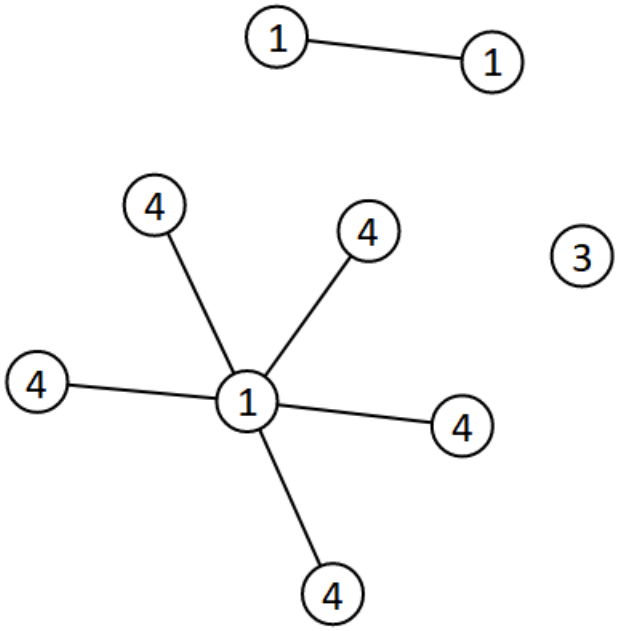
- Awalnya soal ini diniatkan untuk diselesaikan dengan bruteforce, ternyata bisa digunakan linked list, sehingga soal dimodifikasi lagi
- Saat lomba dilaksanakan kami baru tau STL List pada C++ bisa digunakan untuk menyelesaikan soal ini.

Problem J - Jumlah Opat

Author: Jauhar Wibisono
Expected Difficulty: Medium

Salah satu solusi untuk soal ini adalah algoritma *Monte Carlo*. Apabila dipilih sebuah *connected subgraph* secara acak, peluang jumlah nilai *node*-nya habis dibagi 4 adalah 1/4. Dengan kata lain, nilai harapan dari banyak percobaan memilih *connected subgraph* sampai didapatkan *connected subgraph* dengan jumlah nilai habis dibagi 4 adalah 4. Jadi, kita dapat mencoba berbagai *connected subgraph* sampai solusi ditemukan atau sampai kita yakin bahwa tidak terdapat solusi pada graf yang diberikan.

Perlu diperhatikan bahwa *tree* yang tidak memiliki *connected subgraph* dengan jumlah nilai habis dibagi 4 mudah dibuat, contohnya seperti berikut.



Jadi percobaan harus dilakukan secara merata untuk semua *tree* agar *tree* yang memiliki *connected subgraph* dengan jumlah nilai habis dibagi 4 tidak terlewat.

Kode Solusi: https://ideone.com/R5d1dV (solusi random), https://ideone.com/kDxhNV (solusi DP)

Kompleksitas Waktu: $O(N)$

Catatan Panitia: Tree terbesar yang hanya memiliki satu solusi yang ditemukan penulis soal memiliki 7 node, bisakah Anda mencari tree yang lebih besar?