Panduan OSN Informatika berbasis Kurikulum 2017

William Gozali, Alham Fikri Aji

Abstract

Dokumen ini merupakan panduan persiapan Olimpiade Sains Nasional bidang informatika yang dirancang mengikuti kurikulum resmi 2017. Pada dokumen ini terdapat penjelasan singkat untuk tiap bab beserta referensi lanjutan, contoh soal, serta pembahasan.

halo

Contents

Aritmetika dan aturan berhitung

Aritmetika dan aturan berhitung melingkupi:

Aritmetika modular

Memahami konsep operasi modular. Operasi $a \bmod m$, atau biasa disebut "a modulo m" memberikan sisa hasil bagi a oleh m.

Bilangan prima

Bilangan prima adalah bilangan bulat positif yang hanya habis dibagi oleh 1 dan dirinya sendiri. Memahami algoritma-algoritma untuk mengecek bilangan prima, menghasilkan bilangan prima, ataupun melakukan faktorisasi prima.

KPK dan FPB

Mencari KPK dan FPB dengan memanfaatkan faktorisasi prima.

Pigeonhole Principle

Konsep PHP menyatakan bahwa "Jika ada N burung dan M sangkar, dimana $N > M$, maka ada sangkar yang berisi setidaknya 2 ekor burung". Secara matematis, jika ada N burung dan M sangkar, maka ada sangkar yang berisi setidaknya $\lceil \frac{N}{M} \rceil$ ekor burung.

Aturan perkalian dan penjumlahan

Permutasi dan kombinasi

Permutasi dari N adalah seluruh cara pengurutan N objek tersebut. Banyaknya permutasi dari N dinyatakan dalam faktorial, atau $N!$. Rumus faktorial, dinotasikan $N!$ adalah hasil kali dari 1 sampai N, sehingga $N! = 1 \times 2 \times 3 \times \cdots \times N$ Kombinasi adalah banyaknya cara mengambil K buah objek dari N pilihan yang ada, tanpa memerdulikan urutan. Notasi kombinasi ini dinyatakan dalam $C_K^N =$.

Probabilitas

Menghitung kemungkinan terjadinya suatu persistiwa. Secara umum, probabilitas direpresentasikan dalam bentuk $\frac{A}{B}$ dimana A adalah total cara terjadinya peristiwa yang diinginkan dan B adalah total cara terjadinya semua peristiwa. Sebagai contoh, jika terdapat N bola merah dan M bola biru, dan Anda mengambil tepat 2 bola secara acak, berapa kemungkinan bahwa kedua bola berwarna biru? Banyaknya cara mengambil 2 bola biru adalah C_2^N dan banyaknya cara mengambil 2 bola sembarang warna adalah C_2^{N+M}.

Prinsip Inklusi dan Eksklusi

Segitiga Pascal

Segitiga Pascal merupakan susunan dari Koefisien Binomial dalam bentuk segitiga. Nilai dari baris ke-n suku ke-r adalah $A_{ij} = C_r^n$.

Contoh soal:

> Anda diberikan sebuah string S. Ada berapa kata berbeda yang dapat disusun dari huruf-huruf penyusun kata S?
>
> ```
> Contoh 1  : S = 'aba'.
> Jawaban 1 : 3 ('aab', 'aba', 'baa')
>
> Contoh 2  : S = 'MEGAGIGA'
> Jawaban 2 : 3360
> ```

Soal tersebut adalah soal kombinatorik. Mari kita selesaikan contoh 2 terlebih dahulu.

- Terdapat 8 huruf, sehingga banyak kata yang dapat disusun adalah 8!.
- Terdapat 3 huruf 'G' sehingga terdapat 6 kata yang kita anggap berbeda ($G_1G_2G_3, G_1G_3G_2, \ldots, G_3G_2G_1$) yang mana seharusnya keenam kata tersebut merupakan kata yang sama.
- Dengan prinsip Redundansi, maka banyak kata yang dapat disusun mengingat kesamaan kata pada huruf G adalah $\frac{8!}{3!}$.

- Perlu kita perhatikan pula bahwa terdapat 2 huruf A, sehingga dengan cara yang sama akan didapatkan banyak kata yang berbeda adalah $\frac{8!}{(3!\times 2!)} = 3360$.

Maka, solusi secara umum

- Terdapat N huruf, sehingga banyak kata yang dapat kita susun adalah $N!$.
- Apabila terdapat K huruf dengan setiap hurufnya memiliki R_i huruf yang sama, maka dengan prinsip Redundansi banyak kata berbeda yang dapat disusun adalah $\frac{N!}{(R_1!\times R_2!\times R_3!\times \ldots \times R_K!)}$.
- Rumus itulah yang kita kenal dengan Permutasi Elemen Berulang.

Tautan Pendalaman Materi:

- Materi Kombinatorik TLX Training Gate
- Materi Matematika Diskret TLX Training Gate

Soal Latihan

TLX: Faktorisasi Prima

bla

TLX: Penjumlahan Pecahan

bla

TLX: Prima ke-K

bla

TLX: Pasar Rakyat

bla

OSN 2006: Faktorial

bla

Codeforces: Random Teams

bla

Brute Force

Brute-force merupakan suatu strategi penyelesaian masalah dengan mencoba semua kemungkinan. Brute-force menjamin solusi selalu benar, namun biasanya lambat karena menjelajahi semua kemungkinan solusi. Pada OSN, Brute-force berguna untuk menguji kebenaran solusi utama kita. Selain itu, biasanya tiap soal selalu memiliki sub-task yang bisa diselesaikan dengan teknik Brute-force.

Contoh soal:

> Anda diberikan sebuah array A yang terdiri dari N buah bilangan, dan juga bilangan K. Apakah terdapat subset dari bilangan-bilangan tersebut sehingga jumlahan dari elemen subset tersebut sama dengan K? item Bila iya, maka keluarkan "YA". Selain itu keluarkan "TIDAK"
>
> ```
> Contoh  1 : A = [1, 3, 6, 10], K = 14
> Jawaban 1 : "YA" (K = 1 + 3 + 10)
>
> Contoh  2 : A = [1, 3, 6, 10], K = 15
> Jawaban 2 : "TIDAK"
> ```

Solusi soal di atas adalah:

- Untuk setiap elemen, kita memiliki 2 pilihan yaitu memilih elemen tersebut atau tidak memilihnya.
- Kita akan menelusuri semua kemungkinan pilihan.
- Jika jumlahan dari elemen-elemen yang dipilih sama dengan K, maka terdapat solusi.
- Hal ini dapat dengan mudah diimplementasikan secara rekursif.

Mari kita analisa solusi tersebut. Terdapat 2^N kemungkinan pilih-tidak pilih, sehingga kompleksitas solusi adalah $O(2^N)$. Nilai 2^N tumbuh dengan sangat cepat, sehingga solusi ini hanya dapat menyelesaikan soal tersebut jika N kecil.

Tautan Pendalaman Materi:

- Materi Brute-Force TLX Training Gate

Soal Latihan

OSN 2007: Permutasi Ekspresi

> bla

OSN 2012: Kontes Menari

bla

Dynamic Programming

Dynamic Programming (DP) merupakan sebuah teknik dalam strategi penyelesaian masalah. Seperti Greedy, suatu persoalan dapat diselesaikan dengan teknik DP jika solusi optimal dari persoalan dapat ditentukan dari solusi optimal sub-persoalan tersebut. Perbedaannya, pada DP, sub-persoalan tersebut muncul berkali-kali di mana kita menyimpan solusi optimal dari sub-persoalan tersebut dalam tabel. Untuk menguasai DP, dibutuhkan banyak latihan soal. DP hampir selalu muncul dalam OSN (Silahkan lihat contoh-contoh soal) dan sangat direkomendasikan untuk Anda menguasai topik ini.

Contoh soal:

- Diberikan
  jenis koin, masing-masing jenis bernilai
  rupiah.
- Asumsikan terdapat tak hingga koin untuk setiap nominal koin yang ada.
- Tentukan berapa banyaknya minimum koin untuk membayar sebesar
  rupiah!

```
Contoh 1  : M = [1, 6, 10], N = 12
Jawaban 1 : 2 (6 + 6)
```

Soal tersebut adalah soal DP klasik "Coin Change". Mari perhatikan properti berikut:

- Untuk membayar
  rupiah, kita dapat memilih salah satu koin terlebih dahulu.
- Jika nilai koin itu adalah
  , maka sisa uang uang perlu kita bayar adalah
  .
- Perhatikan bahwa penukaran
  merupakan suatu sub-persoalan yang serupa dengan persoalan awalnya. Artinya, cara yang sama untuk menyelesaikan sub-persoalan dapat digunakan.

Bagaimana menyelesaikan soal tersebut secara DP?

- Definisikan sebuah fungsi
  sebagai banyaknya koin minimum yang dibutuhkan untuk membayar
  rupiah.
- Kita dapat mencoba-coba koin yang ingin kita gunakan.

- Jika suatu koin
  digunakan, maka kita membutuhkan
  koin ditambah satu koin
  .
- Atau dapat ditulis
- Pencarian nilai
  dilakukan secara rekursif, kita kembali mencoba-coba koin yang ingin digunakan.
- Fungsi
  ini akan dikunjungi berkali-kali! Sehingga nilai
  perlu kita simpan dalam tabel.

Ini adalah salah satu contoh permasalahan DP klasik. Untuk contoh-contoh klasik lainnnya, bisa dilihat di Materi Contoh DP Klasik TLX Training Gate berikut.

Tautan Pendalaman Materi:

- Materi Perkenalan DP TLX Training Gate
- Materi Studi Kasus DP TLX Training Gate

Soal Latihan

Greedy

merupakan sebuah teknik dalam strategi penyelesaian masalah. Suatu persoalan dapat diselesaikan dengan teknik Greedy jika persoalan tersebut memiliki memiliki properti berikut:

- Solusi optimal dari persoalan dapat ditentukan dari solusi optimal sub-persoalan tersebut.
- Pada setiap sub-persoalan, ada suatu langkah yang bisa dilakukan yang mana langkah tersebut menghasilkan solusi optimal pada sub-persoalan tersebut. Langkah ini disebut juga Greedy Choice.

Contoh soal:

> Diberikan
>
> buah aktivitas.
>
> - Aktivitas ke-
>   dinyatakan dalam
>   .

- Artinya, aktivitas ini dimulai pada waktu
  dan berakhir pada waktu
  .
- Pada setiap satuan waktu, Anda dapat mengikuti paling banyak satu aktivitas.
- Anda ingin mengatur jadwal sedemikian sehingga Anda bisa ikut aktivitas sebanyak mungkin.
  `Contoh  : aktifitas = [<1, 3> , <2, 6>, <5, 7>,  <8, 9>]`
  `Jawaban : [<1, 3>, <5, 7>, <8, 9>]`

Untuk menyelesaikan soal tersebut, kita lihat apakah soal tersebut dapat dijadikan subsoal yang lebih kecil:

- Misalkan kegiatan pertama yang kita ikuti adalah kegiatan ke-
  .
- Kegiatan selanjutnya yang diikuti haruslah memiliki waktu awal
  .
- Lebih jauh lagi, ternyata kita mendapat persoalan yang serupa, hanya saja ukurannya lebih kecil.
- Dengan kata lain, kita memperoleh sub-persoalan.

Untuk setiap sub-persoalan, kita harus memilih sebuah Greedy Choice. Pada soal ini, setidaknya terdapat 3 Greedy choice:

- Memilih aktivitas dengan waktu mulai paling awal.

  pilihan ini tidak tepat, karena bisa jadi ada aktivitas yang mulai lebih awal, tetapi memiliki durasi yang sangat panjang sehingga menyita waktu.

- Memilih aktivitas dengan durasi paling singkat.

  pilihan ini juga tidak tepat. Bisa jadi aktivitas dengan durasi paling singkat ini memotong dua aktivitas lain yang sebenarnya dapat kita ikuti.

- Memilih aktivitas dengan waktu akhir paling awal.

  Dengan memilih aktivitas yang selesai lebih awal, kita mempunyai sisa waktu lebih banyak untuk aktivitas lainnya.

Tautan Pendalaman Materi:

- Materi Greedy TLX Training Gate

Soal Latihan

Pengurutan dan Pencarian

Pengurutan melingkupi teknik-teknik mengurutkan data, seperti:

- **Bubble sort**

  Bubble sort adalah teknik pengurutan data dengan membandingkan suatu elemen dengan elemen yang bersebelahan, dan menukarnya jika urutannya terbalik. Proses ini diulang hingga data terurut.

- **Insertion Sort**

  Insertion sort bekerja dengan menyisipkan elemen pada data satu persatu sehingga hasil penyisipan selalu terurut.

- **Counting Sort**

  Counting sort dilakukan dengan menghitung berapa kemunculan elemen dengan nilai 1, 2, 3, ..., hingga K. Kemudian, angka tersebut ditampilkan lagi secara terurut.

- **Merge Sort dan Quick Sort**

  Ini adalah teknik pengurutan lanjutan. Akan dijelaskan pada bab Divide and Conquer

Sesuai namanya, pencarian adalah proses mencari suatu elemen pada data. Secara umum terdapat 2 teknik pencarian:

-

Linear Search

Linear search adalah proses mencari elemen pada suatu data dengan membandingkan elemen yang ingin dicari pada setiap elemen pada data satu persatu.

-

Binary Search

Jika data yang ada terurut, maka kita dapat mencari lebih efisien. Pada binary search, kita membandingkan elemen yang ingin dicari pada elemen tengah data. Jika elemen tersebut lebih kecil dari elemen tengah pada data, Anda cukup mencari lagi di separuh pertama data tersebut. Jika tidak, kita cari di separuh terakhir. Proses ini dilakukan hingga elemen ditemukan atau data sudah tidak bisa dibagi lagi.

Contoh soal 1:

Anda diberikan sebuah array A yang terdiri dari N buah bilangan. Keluarkan array tersebut dalam kondisi terurut menaik.

```
Contoh  : A = [10, 7, 3, 6]
Jawaban : [3, 6, 7, 10]
```

Contoh soal 2:

Anda diberikan sebuah array A yang terdiri dari N buah bilangan, dan sebuah bilangan D. Tentukan indeks dari bilangan D pada array tersebut. Jika D tidak ditemukan, keluarkan -1

```
Contoh 1  : A = [10, 7, 3, 6] , D = 7
Jawaban 1 : 2

Contoh 1  : A = [10, 7, 3, 6] , D = 9
Jawaban 1 : -1
```

Soal pertama adalah soal pengurutan klasik. Soal ini bisa diselesaikan dengan salah satu teknik pengurutan, misalnya bubble sort. Namun perlu diperhatikan, jika ukuran array cukup besar, maka harus menggunakan teknik pengurutan yang lebih cepat.

Soal kedua adalah soal pencarian. Solusinya adalah dengan melakukan sequential search sebagai berikut:

- Periksa satu per satu dari sepatu pertama, kedua, ketiga, dan seterusnya.
- Jika ditemukan, langsung laporkan.
- Jika sampai akhir belum juga ditemukan, artinya angka yang dicari tidak ada pada daftar.

Tautan Pendalaman Materi:

- Materi Pengurutan dasar: Bubble Sort, Insertion Sort, dan Counting Sort

  catatan: Sorting lanjutan seperti Quick Sort dan Merge Sort akan dipelajari di bab Divide and Conquer

- Materi Pencarian TLX Training Gate

Soal Latihan

Struktur Data Linear

Struktur data linear adalah struktur data yang dibangun pada array. Dalam OSN, terdapat 2 struktur data linear yang masuk dalam kurikulum:

- **Queue**

  Queue adalah struktur data dimana kita dapat memasukkan elemen, dan mengeluarkan elemen sesuai dengan urutan masukknya. Pada Queue, elemen yang pertama dikeluarkan adalah elemen yang pertama kali masuk, mirip seperti antrian dalam dunia nyata.

- **Stack**

  Pada Stack, kita dapat memasukkan elemen dan mengeluarkan elemen yang terakhir kali dimasukkan.

Contoh soal:

> Anda diberikan sebuah string yang terdiri dari kurung "(", ")", dan kurung siku "[", "]". Tentukan apakah string tersebut merupakan barisan kurung yang valid. Barisan kurung yang valid adalah setiap kurung buka memiliki pasangan kurung tutup dengan urutan yang benar.
>
> ```
> Contoh 1  : "([])"
> Jawaban 1 : VALID
>
> Contoh 2  : "([)]"
> Jawaban 2 : TIDAK VALID
> ```

```
Contoh 3  : "(())[][]"
Jawaban 3 : VALID

Contoh 4  : "]["
Jawaban 4 : TIDAK VALID
```

Kita dapat memanfaatkan struktur data Stack. Siapkanlah sebuah stack kosong. Lakukan perulangan terhadap string S. Setiap kali kita menemukan kurung buka "(" ataupun "]", masukkan ke dalam stack. Setiap kali kita menemukan kurung tutup, lihat elemen teratas pada stack dan pastikan pasangan kurungnya tepat.

Tautan Pendalaman Materi:

- Materi Struktur Data Dasar TLX Training Gate

Soal Latihan

Pembahasan Soal Latihan

Aritmetika dan aturan berhitung

TLX: Faktorisasi Prima

Cari seluruh faktor prima dari masukan. Kemudian cetak sesuai keinginan.

TLX: Penjumlahan Pecahan

$E = \frac{(A \times D + B \times C)}{gcd(C,D)}$ dan $F = \frac{C \times D}{gcd(C,D)}$

TLX: Prima ke-K

Gunakan Sieve of Erathosthenes untuk menghasilkan 77.777 bilangan prima pertama. Kemudian cetak sesuai masukan.

TLX: Pasar Rakyat

Hitung KPK dari seluruh masukan

OSN 2006: Faktorial

Menghitung nilai asli dari N! tidak memungkinkan karena terlalu besar. Namun, kita cukup mencari banyaknya faktor 2 dan 5 dari N!, karena 2*5 = 10 (menghasilkan digit 0)

Codeforces: Random Teams

Untuk pasangan teman minimum, distribusikan tim dengan semerata mungkin. Untuk pasangan teman maksimum, distribusikan tim dengan anggota 1 untuk m -1 tim, dan sisanya di tim terakhir. Banyaknya pasangan teman dari suatu tim dengan anggota X orang adalah kombinasi 2 dari X.

Brute Force

OSN 2007: Permutasi Ekspresi

Lakukan simulasi, cari semua permutasi ekspresi yang valid. Karena masukan paling besar adalah 13 digit), maka ada 12 tempat di mana kita dapat menyelipkan operator '+', '-', atau tanpa operator. Sehingga total semua cara yang valid hanyalah 3^{12}. Untuk mengecek hasil ekspresi yang unik, cukup simpan seluruh hasil di array dan lakukan pengurutan.

OSN 2012: Kontes Menari

Karena nilai N hanya 10, kita bisa coba lakukan brute-force di semua permutasi. Untuk setiap permutasi, simulasikan seluruh gerakan untuk mendapatkan total nilai keindahan, dan simpan pada array. Perhitungan kemungkinan rangkaian gerakan yang dapat memukau setiap juri dilakukan dengan melakukan pencarian pada array tersebut.

Dynamic Programming

Greedy

Pengurutan dan Pencarian

Struktur Data Linear