Schematics 2021

EDITORIAL

SCHEMATICS

NPC

NATIONAL PROGRAMMING CONTEST

SPONSORED BY

Editorial Penyisihan Schematics NPC Junior

25 September 2021

Problem	Expected Difficulty	Author
Anak Kembar	Medium	Aldo Yaputra Hartono
Bagi Himpunan	Easy-Medium	Vania Rizky J W
Covid-21	Easy	Reyner Fernaldy
Dihantam Pandemi	Hard	Zydhan Linnar Putra
Eksplorasi Mineral	Medium-Hard	Daniel Sugianto
Fun Quiz	Easy	Aldo Yaputra Hartono
Gala Dinner	Easy	Barhan Akmal Falahudin

Tester

- **Andreas Cendranata**
- **Fadhil Musaad**
- **Haniif Ahmad Jauhari**
- **John Stephanus Peter**
- **Nurlita Dhuha Fatmawati**

Anak Kembar

Author : Aldo Yaputra Hartono

Editorialist : Aldo Yaputra Hartono

Tag : Dynamic Programming

Expected Difficulty : Medium

Subsoal 1 (7 poin)

Observasi : Untuk tiap piring dapat dilakukan rekursi dimana piring tersebut dapat diisi 0 hingga banyak telur yang tersisa.

Kompleksitas: $O(TM^N)$

Subsoal 2 (13 poin)

Observasi : Jika jumlah piring lebih banyak daripada jumlah telur, maka jumlah piring yang digunakan dapat berkurang. Hal ini dikarenakan piringnya identik sehingga *worst case*-nya adalah ketika meletakkan 1 telur pada tiap piring. Dengan kata lain hanya M piring yang terpakai.

Kompleksitas: $O(TM\ min(M,N))$

Subsoal 3 (20 poin)

Observasi 1 : Asumsikan $P(M,N)$ merupakan banyaknya kombinasi peletakan M telur identik pada N piring identik dengan minimal 1 telur pada tiap piring. Mula-mula letakkan 1 telur pada tiap piring sehingga telur yang tersisa sebanyak $M-N$. Karena sudah memenuhi syarat minimal 1 telur tiap piring, maka sisa telur tersebut dapat didistribusikan pada 1 piring, 2 piring, atau bahkan N piring.

$$P(M,N) = \sum_{K=1}^{N} P(M-N,K)$$

Observasi 2 : Asumsikan $Q(M,N)$ merupakan banyaknya kombinasi peletakan M telur identik pada N piring identik dengan minimal 0 telur pada tiap piring. Dikarenakan minimal 0 telur, maka kita bisa meletakkan M telur tersebut pada N piring, $(N-1)$ piring, atau bahkan pada 1 piring saja.

$$Q(M,N) = \sum_{K=1}^{N} P(M,K)$$

Lakukan untuk tiap kasus uji. Keluarkan hasil dari $Q(M,N)$ setelah dimodulo $10^9 + 7$.

Kompleksitas $O(TMN)$

Subsoal 4 (60 poin)

Lakukan *precompute* terlebih dahulu lalu keluarkan hasilnya untuk tiap kasus uji.

Kompleksitas $O(T + MN)$

Bagi Himpunan

Author : Vania Rizky J W

Editorialist : Vania Rizky J W

Tag : Math

Expected Difficulty : Easy-Medium

Untuk semua subsoal berlaku

$1 \leq M \leq N \leq 10^6$

untuk setiap $M > \lceil \frac{N}{2} \rceil$ tidak ada himpunan bagian yang dapat dibentuk

Subsoal 1 (10 poin)

Pada subsoal ini, $1 \leq M \leq N \leq 20$

Setiap elemen mempunyai 2 pilihan, di masukkan ke himpunan bagian atau tidak

Digunakan binary counter dari $00 \ldots 00$ (N angka '0') hingga $11..11$ (N angka '1')

untuk setiap binary counter dicek apakah memenuhi atau tidak

Kompleksitas : $O(N \times 2^N)$

Subsoal 2 (12 poin)

Pada subsoal ini, $N = 2M$

Misalkan barisan M bilangan tersebut secara berurutan adalah

$$x_1, x_2, \ldots, x_{M-1}, x_M$$

dengan syarat untuk setiap $1 \leq i \leq m-1$ berlaku $x_{i+1} - x_i \neq 1$

karena $x_{i+1} \neq x_i$ maka haruslah ada suatu bilangan k dimana

$$x_i < k < x_{i+1}$$

Karena $N = 2M$, maka bilangan yang tidak terpilih ada sebanyak M bilangan

$M-1$ bilangan sudah terpakai untuk mengisi celah x_i dan x_{i+1} untuk setiap $1 \leq i \leq M-1$.

Sehingga ada 1 bilangan tersisa yang dapat ditaruh di celah x_i dan x_{i+1} atau sebelum x_1 atau sesudah x_M

Ada $M + 1$ cara untuk menempatkan bilangan tersebut

Kompleksitas : $O(1)$

Subsoal 3 (48 poin)

Sebenarnya soal ini sama saja dengan menyusun K bola merah dan $N - K$ bola biru dengan syarat tidak ada 2 bola merah yang bersebelahan.

Susun dahulu $N - K$ bola biru

M bola merah akan ditaruh di celah celah setiap bola biru

$$\_B_1\_B_2\_ \cdots \_B_{N-K-1}\_B_{N-K}\_$$

Ada $N - K + 1$ tempat untuk meletakkan K bola merah.

Maka banyaknya cara penempatan adalah

$$\binom{N-K+1}{K}$$

Untuk menghitung nilai kombinasi, kita dapat memanfaatkan segitiga *pascal* dimana

$$\binom{i}{j} = \binom{i-1}{j-1} + \binom{i-1}{j}$$

Kompleksitas : $O(N^2)$.

Subsoal 4 (30 poin)

Sama seperti subsoal 3, hanya saja karena nilai N yang besar, kita tidak bisa menggunakan segitiga *pascal* untuk menghitung nilai kombinasi.

Kita dapat mempercepat perhitungan dengan melakukan prekomputasi untuk nilai bilangan-bilangan faktorial serta nilai *inverse modulo* dari bilangan-bilangan faktorial tersebut. Hal ini dapat dilakukan dalam kompleksitas $O(N)$ atau $O(N \log N)$ tergantung implementasi.

Kompleksitas : $O(N)$ atau $O(N \log N)$

Covid-21

Author : Reyner Fernaldi

Editorialist : Reyner Fernaldi

Tag : Adhoc, String

Expected Difficulty : Easy

Subsoal 1 (10 Poin)

Pada subsoal ini, kita hanya perlu mengecek jika terdapat sebuah *substring* dari virus varian pertama yang terdapat di virus kedua. Jika ya, maka jawabannya adalah *substring* tersebut. Jika tidak, jawabannya adalah vaksin untuk varian virus dengan jumlah infeksi terbanyak.

Kompleksitas: $O(1)$

Subsoal 2 (30 Poin)

Observasi : hanya terdapat 9999 varian vaksin berbeda yang mungkin.

Dengan begitu, untuk setiap varian vaksin, kita dapat melakukan iterasi pada semua varian virus untuk menghitung ada berapa varian virus yang dapat disembuhkan oleh varian vaksin tersebut.

Kompleksitas: $O(N \times K)$ dimana K adalah jumlah varian vaksin berbeda yang mungkin.

Subsoal 3 (60 Poin)

Untuk menyelesaikan subsoal ini, daripada kita menghitung untuk masing-masing varian vaksin, kita dapat menghitung untuk masing-masing varian virus, varian tersebut berkontribusi pada penghitungan varian vaksin yang mana saja.

Kita dapat menginisasi sebuah array dengan ukuran 10000, arr[10000]. Untuk setiap varian virus, dapatkan substring yang terdiri dari 2 sampai 4 karakter. Namun untuk substring yang diawali angka 0 tidak akan di proses. Jadikan substring sebagai indeks pada array arr[]. Kemudian, tambahkan *value* array dengan populasi terinfeksi, sesuai indeks (substringnya).

$$\mathbf{arr}[substring] \mathrel{+}= \ populasi\ infeksi\ varan\ virus$$

Sebagai contoh

B1234 50

Substring dari 1234 adalah 12, 23, 34, 123, 234, 1234. Maka,

$arr[12] \mathrel{+}= 50$

$arr[23] \mathrel{+}= 50$

$arr[34] \mathrel{+}= 50$

$arr[123] \mathrel{+}= 50$

$arr[234] \mathrel{+}= 50$

$arr[1234] \mathrel{+}= 50$

Setelah itu kita dapat mencari jawaban dari nilai arr yang telah dihitung

Kompleksitas: $O(N + K)$ dimana K merupakan jumlah varian vaksin berbeda yang mungkin.

Dihantam Pandemi

Author : Zydhan Linnar Putra

Editorialist : Zydhan Linnar Putra

Tag : Math, Graph, Data Structures

Expected Difficulty : Hard

Subsoal 1 (7 poin)

Observasi : Untuk setiap i $(1 \leq i \leq M)$ berlaku $v_i = u_i + 1$. Graf akan berbentuk seperti garis lurus dengan nomor *node* yang berurutan, misal $(1 \rightarrow 2 \rightarrow 3)$. Oleh karena itu, hanya terdapat satu jalur pulang pergi yang tersedia yaitu jalur yang melewati *node-node* x $(min(a,b) \leq x \leq max(a,b))$. Sehingga *node-node* yang dilewati akan bernilai 2 dan yang lainnya bernilai 0. Solusi ini dapat diimplementasikan menggunakan perulangan biasa.

Kompleksitas: $O(N)$

Subsoal 2 (13 poin)

Subsoal ini mirip dengan subsoal sebelumnya, hanya perlu melakukan *looping*.

Kompleksitas: $O(N)$

Subsoal 3 (17 poin)

Observasi : Graf akan berbentuk *tree* sehingga hanya terdapat tepat satu jalur dari a ke b. Dapat dilakukan *depth-first search* untuk menandai *node-node* yang dilewati pada jalur tersebut. *Node-node* yang dilewati akan bernilai 2 dan yang lainnya bernilai 0.

Kompleksitas $O(N + M)$

Subsoal 4 (35 poin)

Observasi : Graf memiliki *weight* yang sama sehingga untuk mencari jarak terdekat dapat menggunakan *breadth-first search*.

Pertama, cari banyak *shortest path* dari a dan b ke setiap *node* lainnya yang masing-masing berikutnya akan disebut sebagai variabel $cntShortestPathFromStart$ dan $cntShortestPathFromDst$.

Untuk mendapatkan banyak shortest path dari a ke b yang melewati $node_i$ dapat menggunakan rumus berikut:

$$cntShortestPath_i = cntShortestPathFromStart_i * cntShortestPathFromDst_i$$

Hasil $cntShortestPath_i$ adalah jawaban untuk satu kali perjalanan pulang atau pergi sehingga untuk memenuhi aturan soal hasil tersebut harus dikalikan dua.

Berikutnya, untuk mendapatkan rata-rata kunjungan setiap bulannya hanya perlu membagi hasil yang telah dikali dua tersebut dengan total *shortest path* dari a ke b seperti pada penjelasan contoh masukan dan keluaran soal.

Kompleksitas $O(N + M)$

Subsoal 5 (28 poin)

Observasi : Ide penyelesaian sama dengan subsoal 4. Namun karena *weight node* berbeda-beda maka *breadth-first search* harus diganti dengan *dijkstra*.

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

Eksplorasi Mineral

Author : Daniel Sugianto

Editorialist : Muhamad Affan

Tag : Data structures, Dynamic Programming

Expected Difficulty : Medium-Hard

Pertama-tama, mari selesaikan soal ini untuk nilai $D = 1$. Jika D bernilai 1 maka soal ini dapat direduksi menjadi permasalahan untuk mencari luas sub-matriks persegi panjang terbesar yang seluruhnya berisikan nilai 1.

Misalkan matriks awal yang diberikan adalah M.

Kita dapat menggeneralisir solusi ini untuk nilai $D > 1$. Kita dapat membangun matriks $sub_{i,j}[H][L]$ dimana $sub_{i,j}[R][C]$ bernilai 1 jika nilai $M[i][R][C]$ sampai $M[j][R][C]$ semua bernilai 1, dan 0 jika sebaliknya.

Dengan begitu, kita bisa mendapatkan jawaban dengan mencari luas sub-matriks persegi panjang terbesar pada $sub_{i,j}$ untuk $1 \leq i \leq j \leq D$.

Subsoal 1 (30 poin)

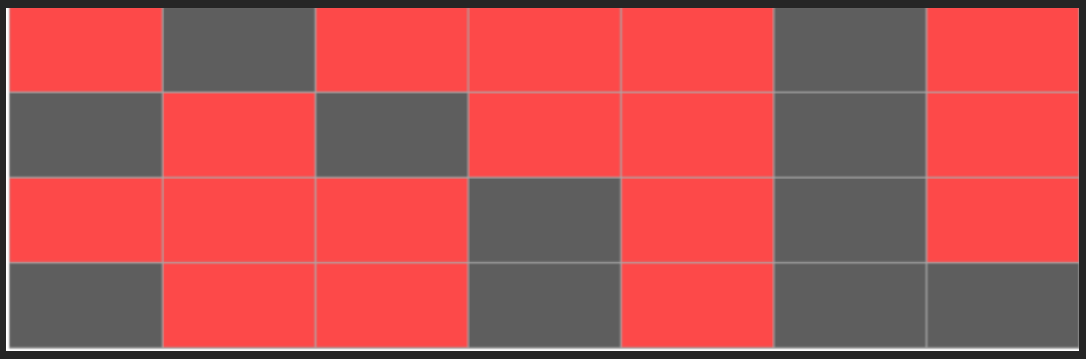
Untuk subsoal ini, kita dapat mencari luas sub-matriks terbesar dengan mengecek semua kemungkinan submatriks yang dapat terbentuk. Terdapat H^2L^2 kemungkinan submatriks. Untuk mengecek apakah suatu submatriks seluruhnya berisikan nilai 1 dapat dilakukan dengan menggunakan *prefix sum* 2 dimensi.

Kompleksitas : $O(D^2H^2L^2)$

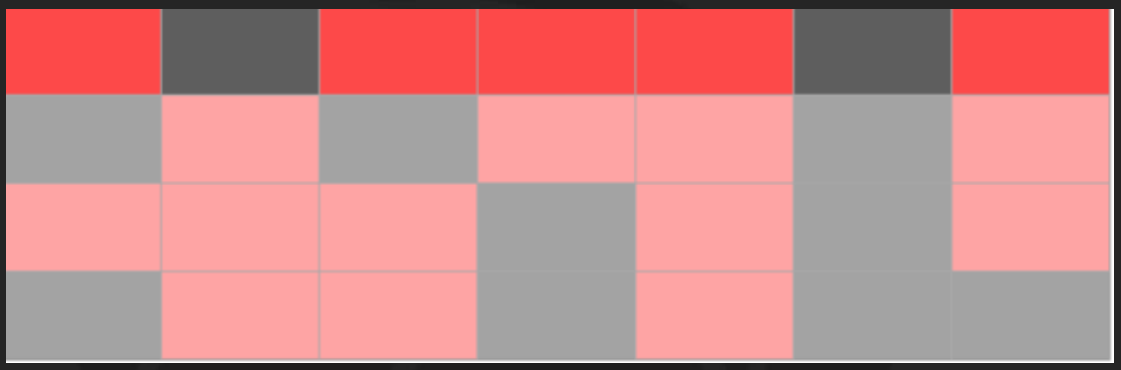
Subsoal 2 (48 poin)

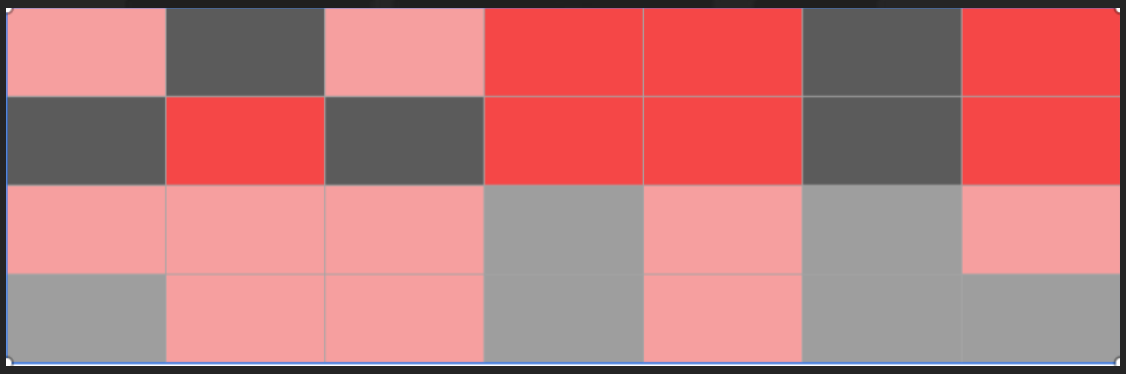
Kita dapat melihat masing-masing baris pada matriks sub sebagai sebuah histogram. Misalkan batang histogram ke-j untuk baris i adalah $Hist[i][j]$. $Hist[i][j]$ bernilai 0 jika $sub[i][j]$ bernilai 0, dan $Hist[i][j]$ bernilai $Hist[i-1][j] + 1$ jika sebaliknya.

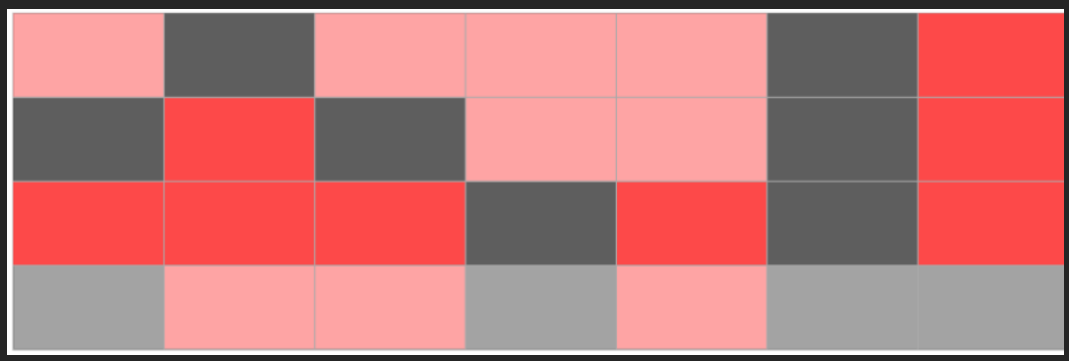
Sebagai contoh, misalkan kotak berwarna merah bernilai 1, serta kotak berwarna hitam bernilai 0. Kita dapat memvisualisasikan matriks seperti ini :

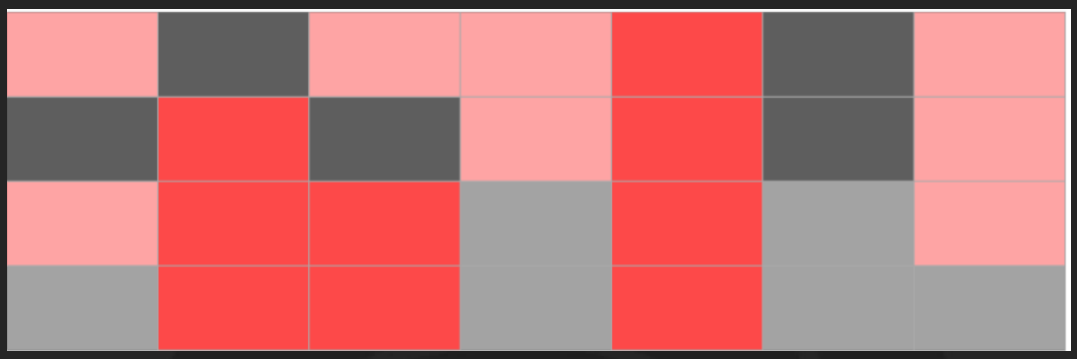


Serta untuk histogram masing-masing baris dapat divisualisasikan seperti ini :









Untuk mencari luas histogram terbesar, kita dapat melakukannya dengan cara *divide and conquer*.

Kita dapat mendefinisikan sebuah fungsi $F(L, R)$ sebagai luas histogram terbesar yang berada di indeks $L - R$. Misalkan X adalah indeks dengan ukuran histogram terkecil. Nilai $F(L, R)$ bisa didapatkan dari nilai maksimal 3 kemungkinan :

- Luas histogram terbesar yang berada di kiri dari indeks X atau $F(L, X - 1)$
- Luas histogram terbesar yang berada di kanan dari indeks X atau $F(X + 1, R)$
- Luas seluruh histogram dengan titik terendah di posisi X atau $Hist[X] * (R - L + 1)$.

$F(L, R)$ dapat dihitung secara rekursif. Serta untuk mencari indeks X dengan efisien, kita dapat membangun *segment tree* dan melakukan operasi *range minimum query*.

Kompleksitas akhir : $O(D^2 HL \log L)$

Subsoal 3 (22 poin)

Idenya adalah untuk setiap histogram i, kita akan menghitung histogram terbesar dengan histogram i menjadi histogram terkecil. Untuk itu, kita dapat mencari indeks terdekat yang berada masing-masing di sebelah kiri dan kanan dari i. Misalkan kedua indeks tersebut masing-masing adalah j_L dan j_R, maka histogram terbesar yang dapat dibentuk adalah $(j_R - j_L - 1) * hist[i]$.

Untuk mendapatkan nilai j_L dan j_R untuk setiap i dapat kita lakukan dengan bantuan struktur data *stack* dengan langkah-langkah berikut :

1. Lakukan iterasi untuk histogram i dari $1 - L$.
2. Jika *stack* dalam keadaan kosong, masukan nilai i kedalam *stack*.
3. Jika *stack* tidak dalam keadaan kosong, lakukan operasi *pop* pada *stack* selama nilai histogram pada *top stack* lebih besar dari nilai histogram i. Untuk setiap histogram yang di-*pop*, kita bisa mendapatkan nilai j_L adalah nilai yang berada tepat dibawah nilai yang di-*pop* tadi, serta nilai j_R adalah i.

Jawaban bisa didapatkan dari nilai luas histogram terbesar untuk masing-masing histogram i.

Kompleksitas : $O(D^2LH)$

Fun Quiz

Author : Aldo Yaputra Hartono

Editorialist : Aldo Yaputra Hartono

Tag : Math

Expected Difficulty : Easy

Subsoal 1 (10 poin)

Observasi 1 : Tiap poin secara unik bisa diperoleh dengan konfigurasi hasil lemparan yang tersusun secara *non-decreasing*.

Observasi 2 : Tiap dadu memiliki S sisi sehingga dapat memunculkan angka 1 hingga S.

Lakukan *nested loop* untuk tiap dadu dan sisinya.

Kompleksitas: $O(SN)$

Subsoal 2 (20 poin)

Observasi : Untuk tiap dadu D yang dibentuk pada observasi subsoal 1, didapatkan pola jumlah sisi tiap dadu pada keseluruhan konfigurasi :

$$D_1 = \frac{(2N+S)(S-1)}{2} + 1$$

$$D_2 = D_1 + (S-1)^2$$

$$D_N = D_{N-1} + (S-1)^2$$

Lakukan penjumlahan dari D_1 hingga D_N.

Kompleksitas: $O(N)$

Subsoal 3 (70 poin)

Observasi 1 : Poin terkecil akan didapat jika semua mata dadu yang muncul bernilai 1. Jika ada N dadu, maka poin terkecilnya adalah $1 \times N = N$.

Observasi 2 : Poin terbesar akan didapat jika semua mata dadu yang muncul bernilai S. Jika ada N dadu, maka poin terkecilnya adalah $S \times N = SN$.

Observasi 3 : Lakukan penjumlahan dari N hingga SN dengan rumus deret aritmatika.

$$\sum_{i=1}^{SN} i - \sum_{j=1}^{N-1} j$$

Jangan lupa keluarkan hasil dalam modulo $10^9 + 7$.

Kompleksitas $O(1)$.

Gala Dinner

Author : Barhan Akmal Falahudin

Editorialist : Barhan Akmal Falahudin

Tag : Math,Greedy

Expected Difficulty : Easy

Subsoal 1 (7 poin)

Observasi: Hanya ada 1 chef dalam Rumah Makan Eva. Oleh karena itu, total kenikmatan maksimal adalah:

$$T = K_1 + K_2 + K_3 + \cdots + K_P$$

Langkah tersebut dapat dilakukan dengan *linear search*.

Kompleksitas: $O(N)$

Subsoal 2 (12 poin)

Observasi: Setiap chef hanya membuat tepat satu makanan. Untuk mendapatkan total kenikmatan maksimal, dilakukan observasi:

1. P adalah jumlah chef yang ada di Rumah Makan Eva.
2. Setiap P makanan pertama haruslah makanan dengan kalori paling minimum dari setiap chef yang berbeda.
3. Kalori makanan di awal dikalikan dengan angka yang yang lebih kecil. Secara matematis dituliskan dalam:

$$T = 1 * K_1 + 2 * K_2 + 3 * K_3 + \cdots + P * K_P$$

4. Makanan di awal dikalikan dengan konstanta yang lebih kecil. Oleh karena itu, akan menguntungkan jika makanan diurutkan secara menaik berdasarkan kalorinya.

Langkah tersebut dapat dilakukan dengan *sorting* dan *linear search*.

Kompleksitas: $O(N \log N)$

Subsoal 3 (81 poin)

Observasi 1 : Untuk mengetahui apakah Pisi dapat memenuhi keinginannya mendapatkan total kenikmatan sebesar M, perlu diketahui total kenikmatan maksimal dari setiap makanan.

Observasi 2: Total kenikmatan makanan yang sedang dimakan adalah jumlah chef berbeda yang makanannya telah dimakan. Oleh karena itu, Untuk mendapatkan nilai kenikamatan maksimum, didapatkan dengan observasi:

1. N adalah jumlah chef yang ada di Rumah Makan Eva.
2. Setiap N makanan pertama haruslah makanan dengan kalori paling minimum dari setiap chef yang berbeda.
3. Kalori makanan di awal dikalikan dengan angka yang yang lebih kecil. Secara matematis dituliskan dalam:

$$T = 1 * K_1 + 2 * K_2 + 3 * K_3 + \cdots + N * K_N$$

4. Oleh karena itu, menguntungkan jika N makanan diurutkan dengan urutan menaik.
5. Setelah itu, makanan selain yang telah termasuk dalam N dijumlahkan, dan dikalikan dengan N dan dijumlahkan dengan total kenikmatan sebelumnya untuk mendapatkan total kenikmatan akhir. Secara matematis dapat dinyatakan dalam:

$$T = T + (K_{N+1} + K_{N+2} + \cdots + K_P) * N$$

6. Dilakukan pengujian apakah total kenikmatan akhir tidak kurang dari M.

Langkah tersebut dapat dilakukan dengan *sorting* dan *linear search*.

Kompleksitas $O(N \log N)$