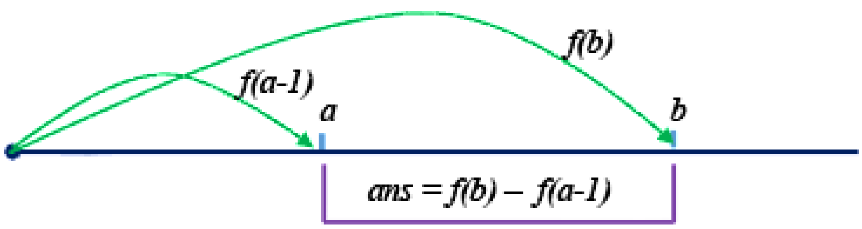
VU34. MÃ SỐ

Giải thuật: Tổng tiền tố.

Nhận xét:

Việc thống kê các đối tượng trong một khoảng liên tục [a, b] thường dựa trên hàm tổng tiền tố `f(x)` xác định số lượng đối tượng thỏa mãn điều kiện cần tìm trong khoảng [0, x], khi đó kết quả cần tìm sẽ là `f(b)-f(a-1)`.



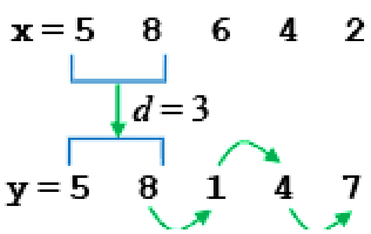
Để tìm các số dạng cấp số cộng chỉ xét các công bội d = 0, 1, 2, . . ., 8, 9,

Dễ dàng thấy rằng tất cả các số có 1 chữ số đều là số dạng cấp số cộng (có tất cả 10 số từ 0 đến 9),

Xét số có đúng k (k > 1) chữ số bắt đầu bằng 1: với các d khác nhau (d = 0, 1, . . ., 9) ta có 10 số khác nhau dạng cấp số cộng, tương tự như vậy với số bắt đầu bằng 2, bằng 3, . . .

Như vậy trong số các số có đúng k chữ số (k > 1) có 90 số dạng cấp số cộng.

Từ 2 chữ số *trái nhất* của số x có k chữ số ta có thể xác định d và từ đó – tìm được số y dạng cấp số cộng có k chữ số và bắt đầu bằng 2 chữ số của số đã cho, ví dụ x = 58642, ta có y = 58147.



Có 2 trường hợp: x ≥ y hoặc x < y.

Cách tính hàm `f(x)` với $x = x_0x_1...x_{k-1}$:

Với k = 1: có x_0+1 số,

Với k > 1: `f(x)` bao gồm:

- Các số dạng cấp số cộng có ít hơn k chữ số: $10+90\times(k-2)$,
- Các số dạng cấp số cộng có k chữ số và chữ số đầu nhỏ hơn x_0: $(x_0-1)\times 10$,

- Các số dạng cấp số cộng có k chữ số bắt đầu bằng x_0 và chữ số tiếp theo nhỏ hơn x_1: x_1 số,
- Thêm 1 số nếu $x \ge y$.

Dữ liệu vào cần lưu trữ dưới dạng xâu (có tới 5 000 chữ số),

Cần có hàm tính xâu `n1-1`, cần lưu ý xóa số 0 không có nghĩa có thể xuất hiện!

Tổ chức dữ liệu: Không cần lưu trữ bảng giá trị phục vụ tính `f(x)` như ở phần lớn các bài toán sử dụng tổng tiền tố vì ở đây `f(x)` có thể dẫn xuất theo công thức.

Độ phức tạp của giải thuật: O(n), trong đó n – độ dài xâu số của dữ liệu vào.

VU46. TÁC ĐỘNG HỖ TRỢ

Giải thuật: Vét cạn, Khai thác khả năng C++.

Nhận xét:

Số lượng các hoán vị cần xét không vượt quá 10! = 3 628 800,

Nếu tìm cách khởi tạo nhanh được các hoán vị, ta có thể dùng phương pháp vét cạn: dẫn xuất hoán vị và tính trực tiếp hiệu ứng hỗ trợ chung,

Để khởi tạo nhanh hoán vị: cần chuyển từ một hoán vị sang hoán vị tiếp sau theo thứ tự từ điển,

Công cụ trong C++: hàm `next_permutation(a+1,a+n+1)` khởi tạo hoán vị tiếp theo theo thứ tự từ điển các phần tử $a_1, a_2, \ldots, a_n$. Nếu không tồn tại hoán vị lớn hơn tiếp theo – trả về giá trị `false`. Độ phức tạp khởi tạo là O(n).

Độ phức tạp của giải thuật: O(n^2).

VV07. XỬ LÝ DỮ LIỆU

Giải thuật: Tìm cầu, chu trình và miền song liên thông của đồ thị.

Nhận xét:

Các servers nối với nhau tạo thành một đồ thị liên thông, trong đó đỉnh là các servers,

Gọi B là tập các đỉnh tạo thành chu trình, nếu ta bỏ một cạnh nào đó trong chu trình thì giữa hai đỉnh bất kỳ thuộc B vẫn có đường đi, như vậy mọi đỉnh trong chu trình có kết nối như nhau không phụ thuộc vào việc có đường truyền nào bị sự cố,

Giả thiết x và y là 2 đỉnh có đường nối trực tiếp, cạnh (x, y) được gọi là *cầu* của đồ thị nếu việc *loại bỏ cạnh này sẽ làm tăng số thành phần liên thô*ng của đồ thị,

Như vậy, sự tồn tại cầu sẽ tác động lên cách xây dựng tập A,

Tìm cầu là bài toán chuẩn trong lý thuyết đồ thị,

Ta có thể tìm tất cả các cạnh là cầu của đồ thị, loại bỏ chúng, đồ thị sẽ phân rã thành các miền liên thông riêng biệt,

Ở mỗi miền liên thông chọn một đỉnh bất kỳ làm đại diện và thay toàn bộ miền liên thông bằng đỉnh đại diện,

Khôi phục lại các cầu đã loại bỏ ta được một cây và các nút lá là những nút cần chọn để xây dựng A,

Như vậy số lượng nút lá là lực lượng của tập A và là một trong số các kết quả cần xác định,

Nếu nút lá là nút đại diện cho miền liên thông chứa p đỉnh, thì với nút lá đó ta có p các chọn đỉnh tham gia vào A,

Số cách xây dựng tập A sẽ là tích các khả năng chọn ở mỗi lá.

Tổ chức dữ liệu: sử dụng các mảng như trong sơ đồ lý thuyết chung xác định cầu của đồ thị.

Độ phức tạp của giải thuật: O($n+m$).

VV29. QUẢN LÝ MỨC

Giải thuật: Cây nhị phân, xử lý bít.

Giá trị lớn nhất trong bộ dữ liệu sẽ đạt tới mức 0 không phụ thuộc và cách bố trí dữ liệu ở các nút lá,

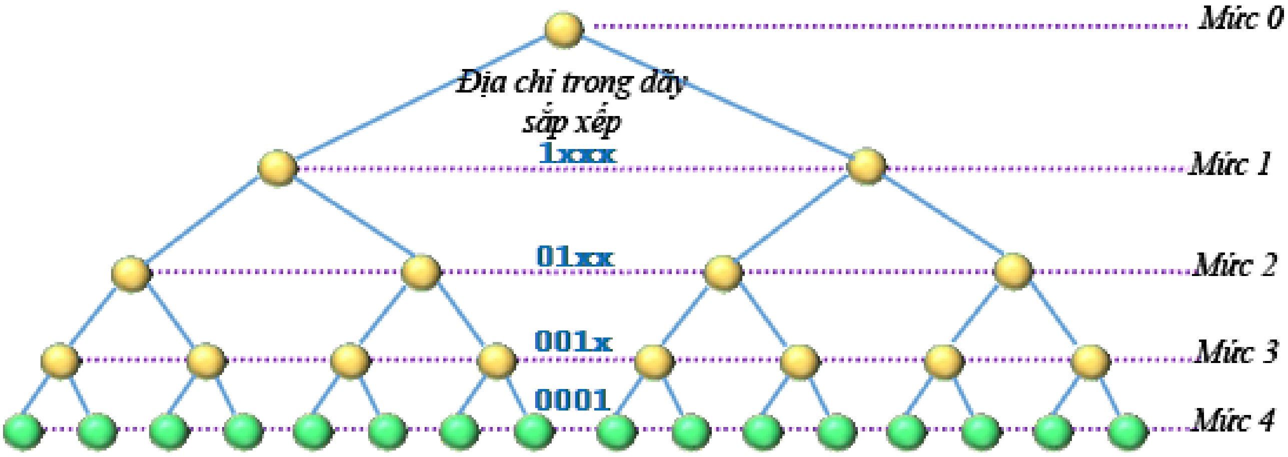
Các phần tử có *giá trị giống nhau* sẽ có *cùng mức nhỏ nhất* có thể đạt được,

Nút gốc chia cây thành 2 cây con, mỗi cây con có 2^{n-1} nút,

Một phần tử muốn chiếm được vị trí gốc cây con này (mức 1) thì phải có không ít hơn 2^{n-1}-1 phần tử nhỏ hơn hoặc bằng nó, nói một cách khác, trong dãy được sắp xếp theo giá trị tăng dần các phần tử của bộ dữ liệu ban đầu, địa chỉ của phần tử gốc cây con phải có bít thứ n-1 bằng 1,

Tương tự như vậy một phần tử muốn chiếm được vị trí mức 2 phải có địa chỉ trong dãy đã sắp xếp có bít thứ n-2 bằng 1,

Tổng quát hóa, ta có một phần tử muốn chiếm được vị trí mức i phải có địa chỉ trong dãy đã sắp xếp có bít thứ n−i bằng 1,



Như vậy toàn bộ quá trình xác định mực được dựa trên việc xử lý địa chỉ của các phần tử có giá trị khác nhau.

VV30. BẢNG SỐ

Giải thuật: Phân tích hoạt động và mô phỏng ô tô mát.

Hai truy vấn có tính chất *thuận – nghịch*, vì vậy có thể dùng kết quả truy vấn loại 1 để kiểm tra kết quả xử lý truy vấn loại 2 và ngược lại,

Mỗi số **v** trong phạm vi từ 1 đến **n** $=2^k$ tương ứng với ô tọa độ (**row**, **col**),

Giá trị **v** có thể xem như địa chỉ tuyến tính của các ô trong bảng, (**row**, **col**) – địa chỉ vật lý của ô,

Để thuận tiện xử lý nên đánh số địa chỉ tuyệt đối bắt đầu từ 0, hàng và cột của bảng cũng đánh số bắt đầu từ 0,

Các phép xử lý đều là biến đổi nhị phân: giá trị các tham số giảm một nửa hoặc tăng gấp đôi sau mỗi bước biến đổi, vì vậy cần chuẩn bị các tham số $b_i = 2^i$, $i = 0 \div 31$, $a_i = 2^{i+1}-1$, $i = 0 \div 30$;

Xử lý truy vấn loại 1:

Đánh số địa chỉ tuyệt đối từ 0: `--v;`

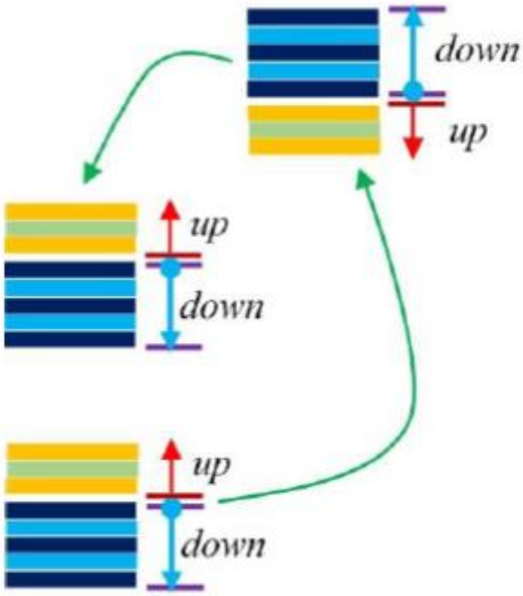
Tính địa chỉ vật lý tương ứng:

- **n** $= 2^k =$ **b[k]**,
- `row =v/n,  col=(v-1)%n;`

Mỗi phép biến đổi gồm 2 bước:

- Bước 1: gấp mép dưới đè lên trên mép trên,
- Bước 2: gấp mép phải đè lên trên mép trái.

Mỗi ô (**i**, **j**) tương ứng với 2 giá trị nguyên: **down** – số lớp *kể từ nó* xuống đáy, **up** – số lớp ở trên nó.



Ở bước 1, khi gấp từ dưới lên trên, nếu ô (**i**, **j**) có **i** < **n/2**, vị trí của nó không thay đổi, số lớp bên trên sẽ tăng thêm một lượng bằng **up+down**, tham số **down** không thay đổi,

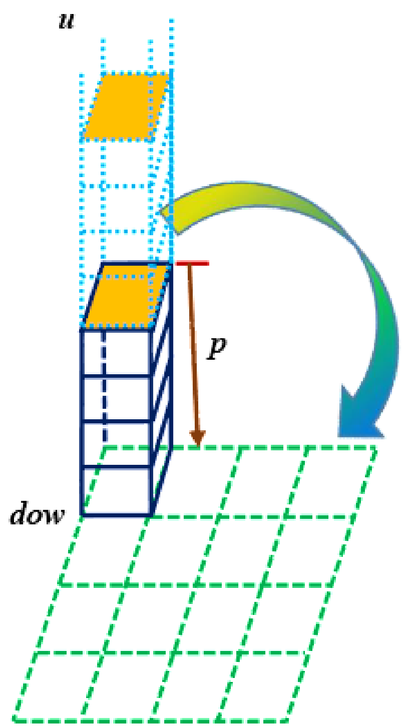
Nếu **i** ≥ **n/2** ô (**i**, **j**) sẽ chuyển vào vị trí ô (**i'**, **j**), trong đó **i'** có các bít nghịch đảo nhị phân với các bít của **i**. Giá trị **up** mới sẽ bằng **down-1**, giá trị **down** mới bằng **2*up+down+1** của các giá trị **up** và **down** cũ.

Số dòng của bảng giảm một nữa.

Tương tự như vậy ở bước 2 với các cột. Số cột sẽ giảm một nửa.

Từ (**row**, **col**) ban đầu việc biến đổi được thực hiện cho đến khi **row=col=0**. Kết quả cần tìm sẽ là giá trị **down**.

Xử lý truy vấn loại 2:



Đặt cột vào vị trí (0, 0),

Mỗi phép biến đổi gồm 2 bước:

- Bước 1: Chia đôi các cột, xét bảng có *số cột gấp đôi*,
- Bước 2: Chia đôi các cột, xét bảng có *số dòng gấp đôi*.

Tham số quản lý ở mỗi ô:

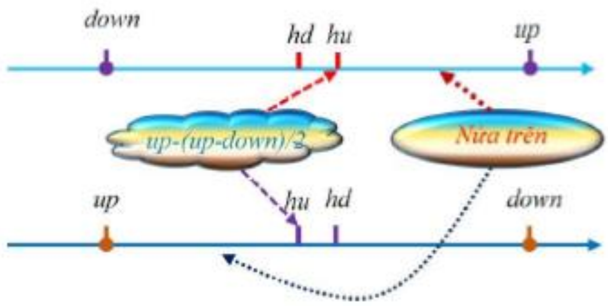
- **down** – số của lớp *ở đáy cột*,
- **up** – số của lớp *trên cùng* của cột.

Giá trị xuất phát:

```
down=0, up=n-1; --p;
row=0, col=0;
```

Khi chia đôi cột, nếu **p** ở nửa dưới thì giữ nguyên **row** và **col**, chỉnh lý lại các biên, nếu **p** ở nữa trên – chỉnh lý **row** và **col** (chỉ sang ô mới), cập nhật các biên **down** và **up**.

Chia đôi cột: Phân biệt 2 trường hợp: **up** > **down** và **up** < **down**.



Kiểm tra **p** thuộc nửa dưới: `(p-v)*(p-hu) > 0`.

Các phép cập nhật **col**, **row**, **up** và **down** – theo phương thức tương tự như khi xử lý truy vấn loại 1.

Điều kiện dừng: **p==down**. Từ tọa độ ô tìm được dễ dàng tính giá trị chưa trong ô.

Độ phức tạp của giải thuật: O($k \times q$).

VV31. KHOẢNG CÁCH HOÀN THIỆN

Giải thuật: Bảng phương án.

Lập bảng **t** tích lũy tổng các ước: t_x – tổng các ước của **x** với $1 \le x \le b$,

Việc tích lũy được tiến hành theo ước số: với mỗi số **d**, các số **2d**, **3d**, . . ., $kd \le b$ đều lớn hơn **d** và có ước là **d**, ước này được tích lũy vào t_{2d}, t_{3d}, . . ., t_{kd}.

Các ước cần xét: **d** = 1, 2, 3, . . ., **b**/2.

Việc tích lũy tổng ước số sẽ có độ phức tạp là:

$$b + b/2 + b/3 + \ldots = b \times \left(1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \ldots\right) \approx b\ln b$$

Kết quả cần tìm: $\mathbf{ans} = \sum_{i=a}^{b} |t_i - i|$.

VV32. ÁO KHOÁC

Giải thuật: Quy hoạch động.

Để lên được lịch trước hết cần xác định với mỗi ngày những áo có thể mặc,

Mỗi ngày có thể có rất nhiều áo phù hợp, nhưng để lên lịch ta chỉ cần ***ghi nhận 3*** trong số đó và gọi là các *áo được chọn*,

Gọi $\texttt{fits}_{i,j}$ là các áo được chon cho ngày thứ $\texttt{i}$,

Lần lượt xét các ngày đã cho, ở ngày $\texttt{i}$ cần tìm $\texttt{j} \in \texttt{fits}_i$ cho phép sử dụng để lập lịch đến ngày $\texttt{i}$, biến $\texttt{dp}_{i,j}$ chứa móc nối tới áo sử dụng ngày $\texttt{i-1}$, giá trị $\texttt{-1}$ chỉ móc nối rỗng,

Giá trị xuất phát: $\texttt{dp}_{0,j}=0 \ \forall \texttt{j} \in \texttt{fits}_0$,

Công thức lặp:

$$\texttt{dp}_{i,j} = \begin{cases} \texttt{k} \in \texttt{fits}_{i-1} \text{ nếu } \texttt{dp}_{i-1,k} \neq \texttt{-1} \text{ và } \texttt{fits}_{i-1,k} \neq \texttt{fits}_{i,j} \\ \texttt{-1} \quad \text{trong trường hợp ngược lại.} \end{cases}$$

Kết quả:

- Không tồn tại lịch nếu $\texttt{dpn-1,j} = \texttt{-1} \ \forall \ \texttt{j}$,
- Trong trường hợp ngược lại: tồn tại lịch phù hợp, dựa theo các móc nối và $\texttt{fits}_i$ ($\texttt{i} = \texttt{n-1} \div 0$) xác định áo cho mỗi ngày.

Xác định các tập áo được chọn:

Mỗi nhiệt độ đã cho được lưu lại dưới dạng bản ghi là nhóm 3 đại lượng:

$\texttt{t}_i \rightarrow (\texttt{t}_i, \ \ 0, \ i)$,

$\texttt{u}_j \rightarrow (\texttt{u}_j, \ -1, \ j)$,

$\texttt{v}_j \rightarrow (\texttt{v}_j, \ \ 1, \ j)$.

Lưu các nhóm 3 này vào trong mảng sự kiện **`events`** và sắp xếp theo thứ tự tăng dần. Cặp bản ghi tương ứng với $\texttt{u}_j$ và $\texttt{v}_j$ xác định khoảng chứa các $\texttt{t}_i$, cho biết áo $\texttt{j}$ phù hợp với ngày $\texttt{i}$.

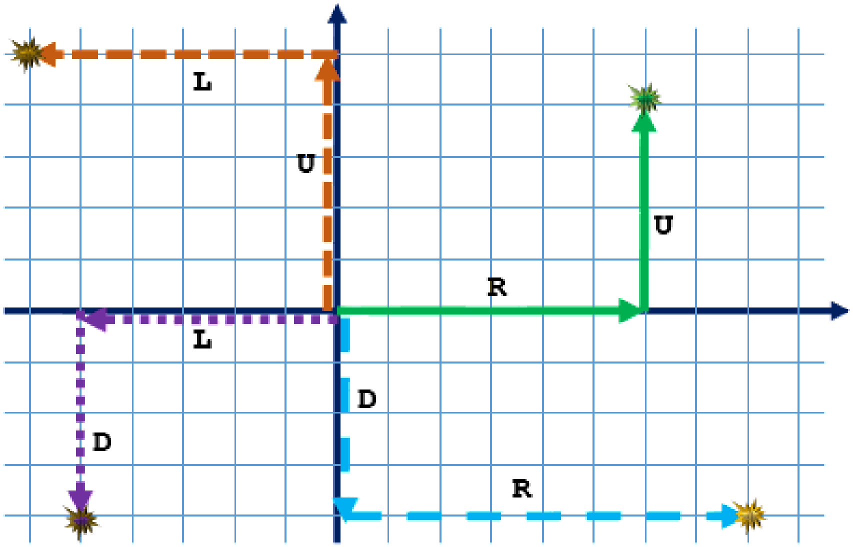
Các cặp $\texttt{u}_j$, $\texttt{v}_j$ được quản lý theo nguyên tắc quản lý cặp ngoặc khi duyệt biểu thức ngoặc.

VV34. KHÓA CHƯƠNG TRÌNH

Giải thuật: Xử lý xâu.

Từ gốc tọa độ (0, 0) tới điểm (**x**, **y**) chỉ cần dùng *tối đa* không quá 2 loại lệnh,

Những lệnh còn lại: khóa ngay từ đầu.



Với những lệnh cần dùng: mỗi lần thực hiện giá trị tuyệt đối của tọa độ tương ứng giảm 1.

Quá trình xử lý kết thúc khi có tọa độ kết quả là (0, 0) hoặc khi duyệt hết xâu **s**.

Cần đưa ra kết quả -1 khi sau khi duyệt hết xâu **s** ta vẫn không nhận được tọa độ (0, 0).

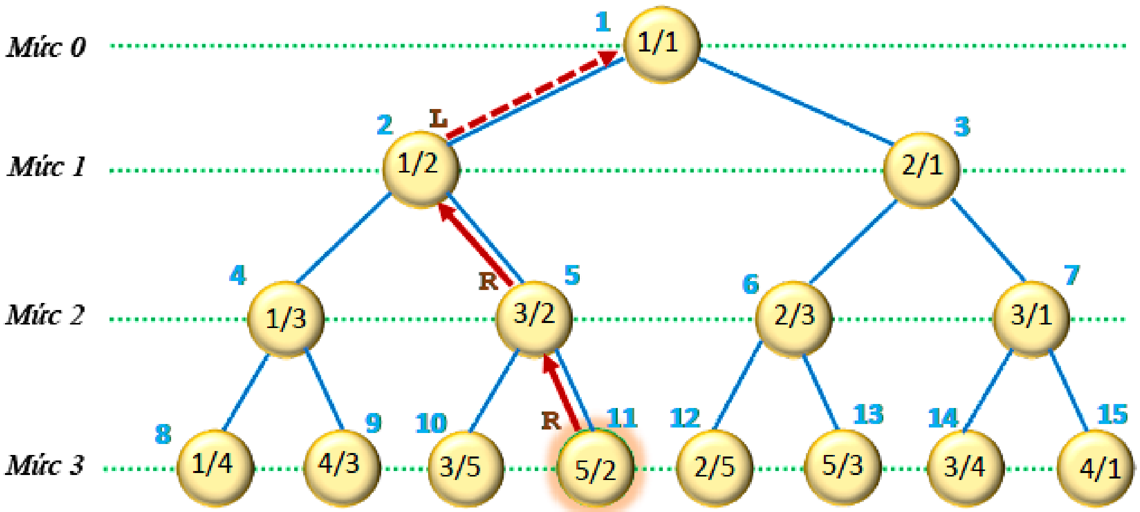
Độ phức tạp của giải thuật: O(*n*).

VV35. THỨ TỰ

Giải thuật: Xử lý bít.

Ta thấy mức **i** có 2^i nút,

Một nút có thứ tự là **k** thì 2 nút con trái và phải của nó có số thứ tự tương ứng là **2k** và **2k+1**. Số thứ tự nút cha của nó là **k/2**.



Xét nút chứa phân số (**p/q**). Nếu **p** > **q** thì nút này là nút con trái của nút cha trực tiếp, trong trường hợp ngược lại – là nút con phải,

Một nút có **p** = 1 thì nó và các nút cha của nó trên đường di chuyển về gốc đều là nút con trái,

Một nút có **q** = 1 thì nó và các nút cha của nó trên đường di chuyển về gốc đều là nút con phải.

Xét đường đi từ nút chứa phân số (**p/q**) về nút có **p** = 1 hoặc **q** = 1, đường đi này tương ứng với xâu **s** chỉ chứa các ký tự thuộc tập {**L**, **R**}. Dựa vào quan hệ so sánh của **p** và **q** ta dễ dàng xác định được xâu **s**.

Ví dụ, với **p** = 5, **q** = 2 ta có:

$$(5, 2) \longrightarrow (3, 2) \longrightarrow (1, 2)$$

$$s = \quad R \quad\quad R \quad \Longrightarrow \quad s = 11$$

Để tiện xử lý, ta thay **L** bằng **0** và **R** bằng **1**.

Bằng cách duyệt đường đi từ cuối về đầu xâu **s** ta dễ dàng tính được số thứ tự của phần tử. Lưu ý là phần còn lại của đường đi về nút gốc không được lưu trữ tường minh trong **s**. Nếu việc xác định **s** dừng lại ở nút (**1/u**) thì nút đó có số thứ tự là 2^{u-1}, nếu nút chứa giá trị (**v/1**) thì số thứ tự là 2^v-1.

Tương tự như vậy với việc xác định **p**, **q** theo số thứ tự **n**. Nếu **n** là chẵn thì nút đang xét là nút con trái, trong trường hợp ngược lại – nút con phải. Dựa vào đường đi ta dễ dàng chỉnh lý **p** và **q**.

Tổ chức dữ liệu:

Để thuận tiện xử lý ta lưu đường đi dưới dạng xâu các ký tự 0 và 1. Có thể vòng tránh việc sử dụng xâu này, nhưng điều đó sẽ làm chương trình phức tạp lên đôi chút một cách không cần thiết. Trong trường hợp xấu nhất độ dài xâu **s** cũng không vượt quá 64.

VV38. A+B

Giải thuật: Phân tích tình huống lô gic, Kỹ thuật tổ chức dữ liệu.

Tư tưởng chung của giải thuật rất đơn giản:

- Tính $t = a_i + a_j$ với $i = 1 \div n$, $j = 1 \div n$, $i \neq j$,
- Với mỗi t tính được kiểm tra sự tồn tại k sao cho $k \neq i$, $k \neq j$ và $a_k = t$.

Độ phức tạp của giải thuật sẽ là $O(n^3)$, *vượt quá giới hạn thời gian* cho phép với n đủ lớn.

Cần giảm độ phức tạp của giải thuật:

- Giảm số lần cần tính giá trị t,
- Giảm thời gian tìm kiếm k với mỗi t.

Ta thấy, nếu nhóm 3 (i, j, k) thỏa mãn điều kiện cần tìm thì nhóm 3 (j, i, k) cũng thỏa mãn.

Với mỗi nhóm 3 (i, j, k) cần tìm có 4 trường hợp có thể xảy ra:

- Cả 2 số a_i và a_j đều dương,
- Cả 2 số a_i và a_j đều âm,
- Có một số nguyên và một số âm,
- Có ít nhất một trong 2 số a_i, a_j bằng 0.

Việc phân biệt các trường hợp sẽ cho phép tổ chức dữ liệu một cách thích hợp và làm giảm đáng kể độ phức tạp của giải thuật.

Không có yêu cầu phải đưa ra các chỉ số i, j, k nên ta chỉ cần lưu các giá trị khác nhau của phần tử thuộc dãy cùng với tần số xuất hiện và lưu trữ ở 2 phiên bản: một phiên bản *các giá trị khác nhau* của dãy, phục vụ tính t và một phiên bản lưu trữ *tần số xuất hiện,* phục vụ kiểm tra, xác định sự tồn tại của k và cập nhật kết quả.

Số lượng phần tử 0 được lưu trữ riêng.

Gọi a_i là tần số xuất hiện giá trị $i > 0$ trong dãy ban đầu, va_j là phần tử thứ j trong dãy các giá trị dương khác nhau, sắp xếp theo thứ tự tăng dần, b_i là tần số xuất hiện giá trị $-i$ ($i > 0$) trong dãy ban đầu, ba_j là phần tử thứ j trong dãy giá trị giá trị tuyệt đối các phần tử âm khác nhau, sắp xếp theo thứ tự tăng dần, zr – số lượng phần tử bằng 0.

Xét các phần tử dương:

- Xét $t = va_i + va_j$, $j \geq i$, $t = va_j + va_j$ tồn tại khi và chỉ khi $a_j > 1$,
- Nếu $a_t \neq 0 \rightarrow$ tồn tại k và số lượng nhóm 3 mới tìm được sẽ là $a_i \times a_j \times a_k \times 2$.

Xử lý tương tự như vậy với các phần tử âm, Trường hợp một số hạng dương và một số hạng âm: xét mọi $\mathbf{t} = \mathbf{va_i} + \mathbf{vb_j}$ và xử lý tượng như trên,

Trường hợp mộtsố hạng bằng 0: (xẩy ra khi **zr** > 0), **t** bằng $\mathbf{va_i}$ hoặc $\mathbf{vb_j}$, **k** chỉ tồn tại khi $\mathbf{va_i}$ hoặc $\mathbf{vb_j}$ có tần số xuất hiện lớn hơn 1, khi đó số lượng nhóm 3 mới sẽ là $\mathbf{zr \times a[va_i] \times 2}$ hoặc $\mathbf{zr \times b[vb_j] \times 2}$.

Trường hợp **t** = 0 – chỉ cần xét khi **zr**>2, số lượng nhóm 3 mới sẽ là **zr×(zr-1)**.

VW04. THỬ NGHIỆM

Giải thuật: Kiến thức vật lý, Phương trình bậc 2.

Với chuyển động nhanh dần đều hoặc chậm dận đều ta có các công thức:

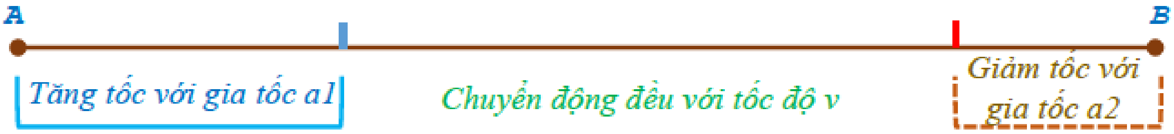
Tốc độ = Gia tốc × Thời gian v = at → $t = \frac{v}{a}$,

Quảng đường đi được từ khi tốc độ ban đầu bằng 0 cho đến khi đạt tốc độ v là

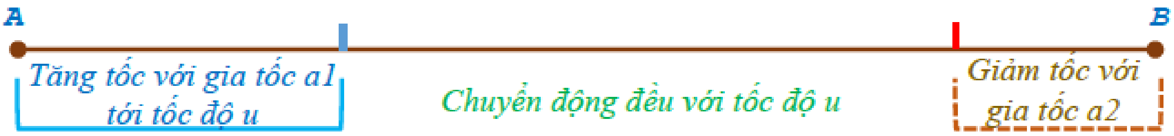
$$s = \frac{at^2}{2} = \frac{av^2}{2a^2} = \frac{v^2}{2a}$$

Có 2 trường hợp xẩy ra:

Tàu có thể đạt tốc độ **v** và *thời gian chạy với tốc độ không đổi **v** lớn hơn hoặc bằng **t***, như vậy thời gian thử nghiệm ngắn nhất sẽ tương ứng *với tốc độ tối đa là **v***:



Tàu cần đạt tốc độ tối đa **u** < **v** và *thời gian chạy với tốc độ không đổi **u** lớn hơn hoặc bằng **t***, như vậy thời gian thử nghiệm ngắn nhất sẽ tương ứng *với tốc độ tối đa là **u***:



Trong trường hợp này ta có phương trình ẩn số u:

$$\frac{u^2}{2a1} + tu + \frac{u^2}{2a2} = d$$

Phương trình bậc 2 này có một nghiệm dương và một nghiệm âm. Giá trị cần tìm tương ứng với nghiệm dương.

Thời gian cần tìm sẽ là tổng thời gian đi trên 3 đoạn.

Độ phức tạp của giải thuật: O(1).

VW10. HỢP TÁC

Giải thuật: Phân tích tình huống lô gic.

Đầu tiên đáp ứng nguyện vọng những người chỉ muốn làm việc với người cùng đoàn, khi đó số lượng người không thỏa mãn nguyện vọng cá nhân sẽ là **a%2+c%2**.

Nếu **b** = **d** – ghép những người này theo nguyện vọng.

Xét trường hợp **b** > **d**:

- Sẽ có **d** cặp hài lòng với cách ghép nhóm,
- Còn lại **b-d** người ở đoàn Rạng Đông,
- Tổng số người chưa được ghép cặp và có khả năng sẽ không hài lòng sẽ là

  a%2+c%2+b-d,

- Nếu **c** lẻ thì ghép người đó với một trong số **b-d** người còn lại trong đoàn Rạng Đông, có cặp trong đó chỉ một người không hài lòng, tổng số người có khả năng sẽ không hài lòng giảm một,
- Số người còn lại chưa được ghép cặp là **rm=b-d-c%2**,
- Nếu **a** là số lẻ và **rm** > 0 thì ghép một người còn dư với một trong só **rm** người cùng đoàn muốn là việc với đoàn khác ta được một cặp trong đó chỉ có một người không vui, số người không hài lòng sẽ giảm một và đó là kết quả tốt nhất có thể!

Trường hợp **b** < **d**: xử lý tương tự.

Độ phức tạp của giải thuật: O(1).

VW15. TINH DẦU OẢI HƯƠNG

Giải thuật: Phân tích hoạt động ô tô mát.

Tính số giọt tinh dầu dò rỉ trong một chu kỳ tích áp:

$$\texttt{drops} = \sum_{i=0}^{n-1}\left(\frac{m}{p_i}\right) \quad (\textit{phép chia nguyên})$$

Số chu kỳ tích áp: `pr = t/m`,

Số tinh dầu dò rỉ ở phần cuối thời gian trước khi sửa máy: $\texttt{r} = \sum_{i=0}^{n-1}(t\%m)/p_i$.

Từ các tham số này dễ dàng dẫn xuất kết quả cần tìm.

Độ phức tạp của giải thuật: O(*n*).

Giải thuật: Tổng tiền tố, Tìm kiếm nhị phân..

Cố định một chỉ số **i** và từ đó – kéo theo việc cố định giá trị a_i, ta xác định cách tính $\sum_j (i-j)|a_i - b_j| = \sum_{j|a_i \ge b_j} (i-j)(a_i - b_j) + \sum_{j|a_i < b_j} (i-j)(b_j - a_i)$

$$\sum_{j|a_i \ge b_j} (i \times a_i - i \times b_j - j \times a_i + j \times b_j) \quad (*)$$

$$i \times a_i \sum_{j|a_i \ge b_j} 1 - i \times \sum_{j|a_i \ge b_j} b_j - a_i \times \sum_{j|a_i \ge b_j} j + \sum_{j|a_i \ge b_j} j \times b_j$$

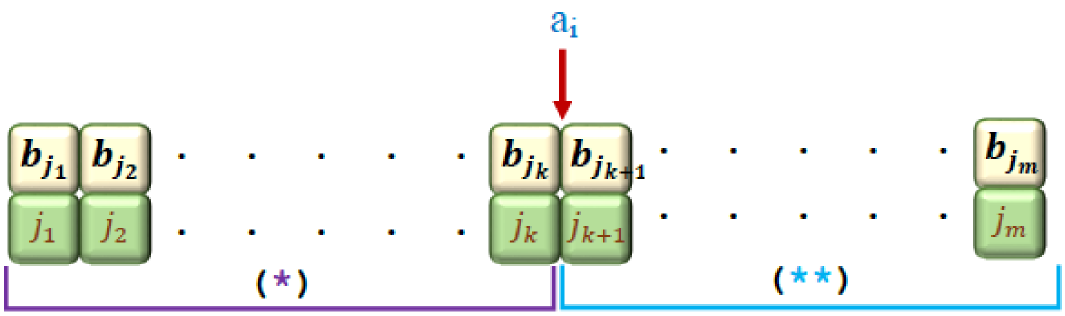
Đổi dấu: **+** → **–**
– → **+**

$$\sum_{j|a_i < b_j} (-i \times a_i + i \times b_j + j \times a_i - j \times b_j)$$

$$-i \times a_i \sum_{j|a_i < b_j} 1 + i \times \sum_{j|a_i < b_j} b_j + a_i \times \sum_{j|a_i < b_j} j - \sum_{j|a_i \ge b_j} j \times \quad (**)$$

Để tính các tổng trên ta cần lưu mảng thứ 2 (mảng **B**) dưới dạng các cặp (b_j, j), sắp xếp chúng theo thứ tự tăng dần và, với mảng đã sắp xếp, chuẩn bị các mảng tổng tiền tố với j, b_j và $j \times b_j$.

Bằng phương pháp tìm kiếm nhị phân, với mỗi a_i có thể xác định được **k** chia mảng chứa (b_j, j) thành hai phần, mỗi phần ứng với một công thức đã nêu.



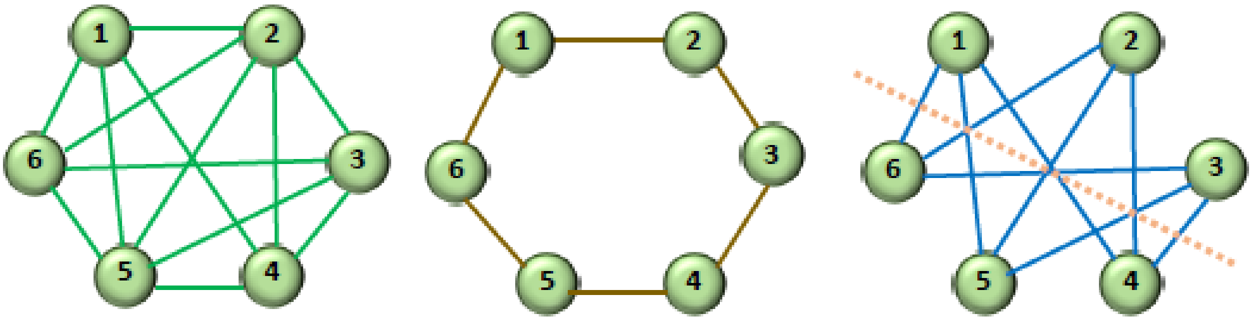
VW20. BẢN ĐỒ DU LỊCH

Giải thuật: Chu trình Hamilton.

Từ lý thuyết đồ thị, với n đỉnh có 3 trường hợp có thể xây dựng một đồ thị m cạnh theo các tính chất đã nêu:

- Đồ thị đầy đủ: $m = \frac{n \times (n-1)}{2}$;
- Một chu trình: m = n,
- Đồ thị 2 phía đầy đủ đối xứng: $m = (\frac{n}{2})^2$.

Ví dụ, với n = 6, ta có 3 đồ thị thỏa mãn yêu cầu đã nêu:



$m = \frac{n \times (n-1)}{2}$ $m = n$ $m == (\frac{n}{2})^2$

Nếu m không thỏa mãn một trong 3 điều kiện trên thì không tồn tại đồ thị thỏa mãn yêu cầu đầu bài.

Việc dẫn xuất các đường đi là không khó khăn. Ở trường hợp thứ 3, các đỉnh từ 1 đến $\frac{n}{2}$ được xếp vào phía thứ nhất, các đỉnh còn lại – phía thứ 2.

Độ phức tạp của giải thuật: $O(n^2)$.

VW21. DÃY CON

Giải thuật: Quy hoạch động.

Quá trình giải bài toán được chia thành 2 bước :

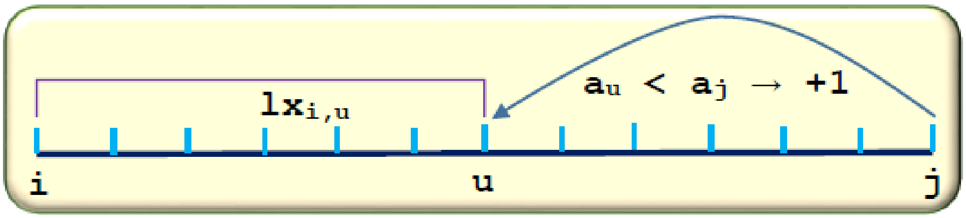
- Bước 1: Tính $\mathbf{lx}_{i,j}$ – độ dài lớn nhất của dãy con tăng dần từ vị trí **i** đến vị trí **j** và có *sự tham gia của phần tử* a_j,
- Bước 2: Tính $\mathbf{f}_{k,r}$ – số lượng dãy con tăng dần có độ dài lớn hơn hoặc bằng **k** trong **r** phần tử đầu tiên của dãy **A**.

Việc tính $\mathbf{lx}_{i,j}$ được thực hiện bằng phương pháp quy hoạch động.

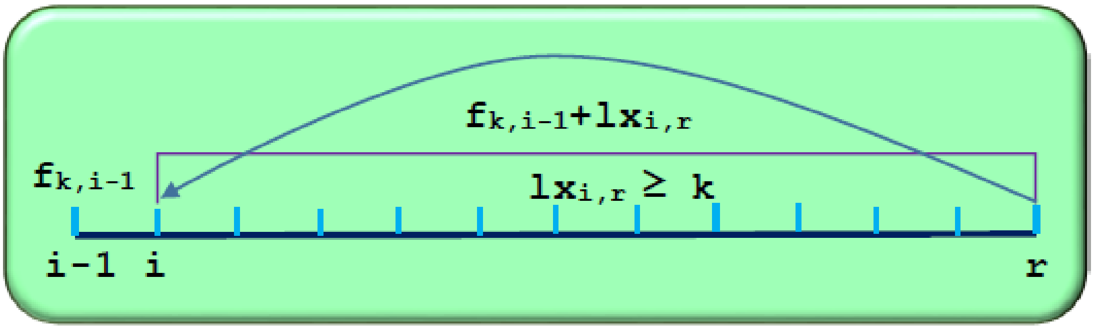
Gọi **m** là độ dài của đoạn cần tính. Tồn tại giải thuật tính $\mathbf{lx}_{i,j}$ với độ phức tạp O($m$ln$m$), nhưng với **m** bé (**m** ≤ 300), để đơn giản lập trình ta có thể tính theo giải thuật độ phức tạp O(m^2):

Khởi tạo: $\mathbf{lx}_{i,j} = 1$,

Công thức lặp: $\mathbf{lx}_{i,j} = max\{\mathbf{lx}_{i,j}, \mathbf{l}_{i,u} + 1\}$ nếu $a_u < a_j$, $u = i, i+1, \ldots j-1$.



Tính $\mathbf{f}_{k,r}$:



Nếu chưa xét đoạn từ **i** đến **r** (**i** < **r**) thì $\mathbf{f}_{k,r} = max\{\mathbf{f}_{k,r}, \mathbf{f}_{k,i-1}\}$.

Xét sự tham gia của đoạn từ vị trí **i** đến vị trí **r**: Nếu $\mathbf{lx}_{i,r} \geq \mathbf{k}$ thì giá trị cần tìm *có thể* là $\mathbf{f}_{k,i-1}+\mathbf{lx}_{i,r}$ nếu nó lớn hơn $\mathbf{f}_{k,r}$ đang có.

Cần duyệt với mọi **i** thay đổi từ 1 đến **n** và với mỗi **i** – duyệt **r** từ **i** đến **n**.

Kết quả: Với mỗi **k** kết quả cần tìm là giá trị $\mathbf{f}_{k,n}$.