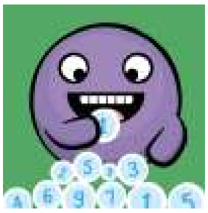
BÀI GIẢNG VỀ KỸ THUẬT LẬP TRÌNH VỚI C++

06 THIẾT KẾ THUẬT TOÁN THEO GIẢI THUẬT THAM ĂN- GREEDY METHOD

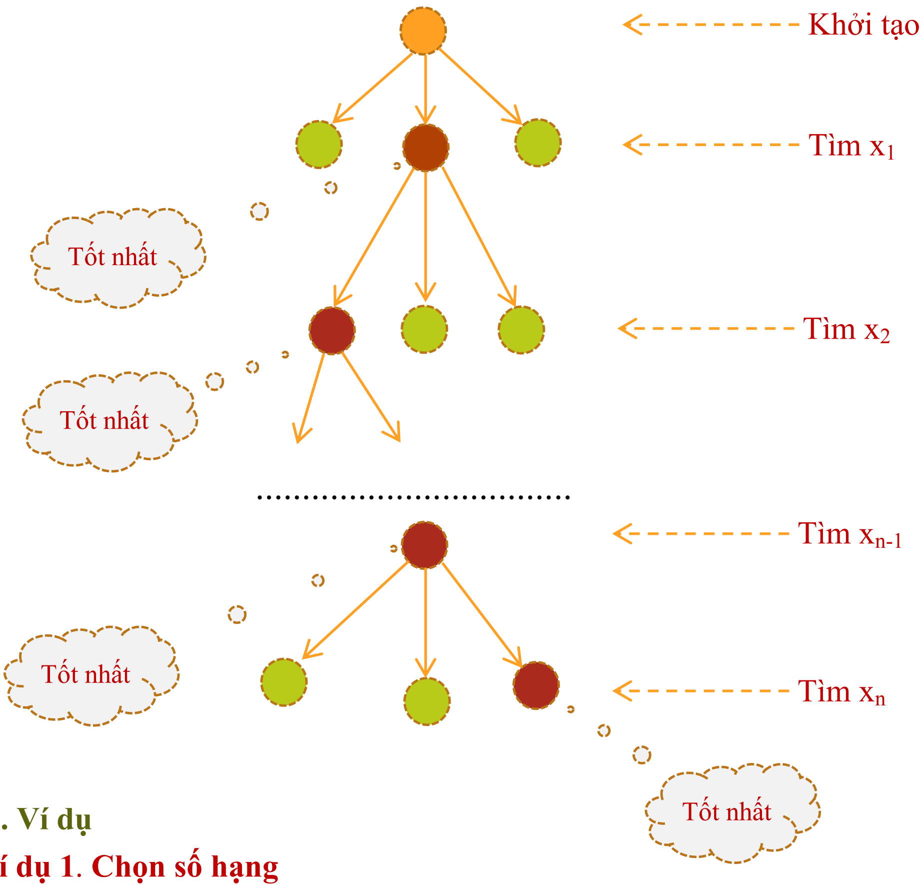


A. LÝ THUYẾT

I. Tư tưởng

Giả sử để tìm lời giải tối ưu cho bài toán P ta cần thực hiện n bước, bước i ta cần tìm thành phần x_i. Nghĩa là nghiệm của bài toán P có thể biểu diễn thành bộ $(x_1, x_2, .., x_n)$. Trong giải thuật tham, để tìm thành phần x_i, cần đưa ra một "***tiêu chí để đánh giá độ tốt***", dựa vào tiêu chí này, ta chọn x_i tốt nhất có thể. Khi xây dựng xong x_n, ta được bộ nghiệm $(x_1, x_2, .., x_n)$ theo chiến lược tham.

Cây chiến lược tham - GREEDY METHOD TREE.



II. Ví dụ

Ví dụ 1. Chọn số hạng

Cho dãy số nguyên $a_1, a_2, .., a_n$ và số tự nhiên k ($k \leq n$). Hãy chọn k số hạng trong dãy để tổng k số hạng được chọn là lớn nhất.

Phân tích chiến lược:

Đây là một bài toán đơn giản nhưng lời giải của nó mô tả một chiến lược "tham". Ta thực hiện k bước, mỗi bước ta chọn số hạng lớn nhất trong các số hạng chưa được chọn.

- Tiêu chí tham đó là: **Chọn số hạng có giá trị lớn nhất trong các số hạng chưa được chọn**.
- Với chiến lược này, ta được nghiệm đúng của bài toán.

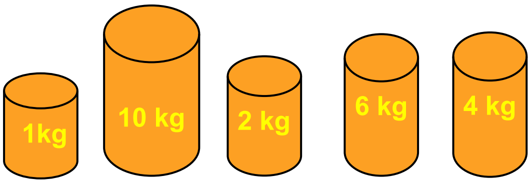
Thực hiện:

Bước 1. Sắp xếp các số hạng theo thứ tự tăng dần.

Bước 2. Chọn k số hạng cuối.

Ví dụ 2. Xếp đồ vào giỏ – 1

Có n đồ vật có khối lượng $a_1, a_2, .., a_n$ (kg) và một cái giỏ chứa không quá M (kg). Hãy tìm cách xếp các đồ vật này vào giỏ sao cho tổng khối lượng các đồ vật không quá M và số đồ vật được xếp là nhiều nhất.



M = 7kg

➢ **Tiêu chí tham?**

Chọn vật có khối lượng nhỏ nhất (chưa bỏ) để bỏ vào giỏ (nếu có thể bỏ được).

➢ **Thực hiện**

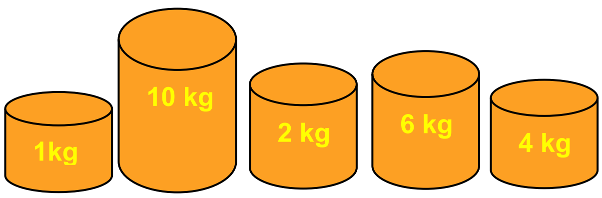
```
sort(a+1, a+1+n)
int kq, s, i;
kq = s = 0;
i = 1;
while(i <= n && s + a[i] <= M){
    kq++;
    s += a[i];
    i++;
}
cout<<kq;
```

➢ **Tính đúng?**

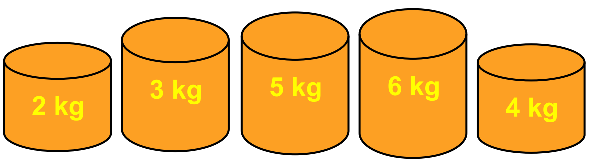
Với chiến thuật tham trên, ta thu được lời giải đúng.

Ví dụ 3. Xếp đồ vào giỏ – 2

Có n đồ vật có khối lượng $a_1, a_2, .., a_n$ (kg) và hai cái giỏ, mỗi giỏ chứa không quá M (kg). Hãy tìm cách xếp các đồ vật này vào hai giỏ sao cho tổng khối lượng các đồ vật trong mỗi giỏ không quá M và số đồ vật được xếp là nhiều nhất.



M = 8kg



M = 8kg

- **Phương án tham**

  Chọn các đồ vật từ khối lượng từ nhỏ đến lớn.
  - Lần lượt bỏ vào ***giỏ thứ nhất*** đến khi không bỏ được nữa, sau đó:
  - Lần lượt bỏ vào ***giỏ thứ hai*** đến khi không bỏ được nữa.

  Với ví dụ trên, các vật sau khi sắp xếp đó là:

  1, 2, 2, 3, 4, 4, 5, 6, 6, 10.

  ➔ Chọn các vật khối lượng: 1, 2, 2, 3 bỏ vào giỏ 1:

  ➔ Chọn các vật khối lượng: 4, 4 bỏ vào giỏ 2.

  Tổng số vật được bỏ vào cả hai giỏ là 6.

- **Tính đúng**

  Ta nhận thấy với ví dụ trên, số đồ vật được bỏ nhiều nhất bằng 6 là một lời giải đúng.

  Nhưng chiến thuật tham này không đúng cho mọi trường hợp.

  Chẳng hạn, xét ví dụ khác:

  Có 8 đồ vật có khối lượng: 1, 2, 3, 4, 4, 5, 5, 10. Hai giỏ, M = 13.

  Với chiến thuật trên:
  - Giỏ 1 bỏ các vật: 1, 2, 3, 4.
  - Giỏ 2 bỏ các vật: 4, 5.

    → Số vật bỏ vào được là 6.

  ➔ **Cách sắp xếp tối ưu**
  - Giỏ 1 bỏ các vật: 1, 2, 4, 5.
  - Giỏ 2 bỏ các vật: 3, 4, 5.

    → Số vật bỏ vào được là 7.

Chú ý:

- **Khi giải một bài toán theo chiến lược tham có thể cho lời giải tối ưu không đúng. Điều này tùy thuộc vào tiêu chí tham và bài toán đó có thể giải bằng giải thuật tham hay không?**
- **Trong chiến lược tham, mỗi bước ta tìm phương án tối ưu – gọi là tối ưu cục bộ (*local optimal*) để gộp lại thành một phương án tổng thể và phương án tổng thể này chưa chắc là phương án tối ưu toàn cục (*global optimal*).**
- **Toán học hiện đại chứng minh được những bài toán có cấu trúc matroid thì có phương án tham để giải đúng các bài toán này.**

B. BÀI TẬP

1☼. Bỏ đồ vào giỏ

Có n đồ vật có khối lượng $a_1, a_2, .., a_n$ (kg) và một cái giỏ chứa không quá M (kg). Hãy tìm cách xếp các đồ vật này vào giỏ sao cho:

- Tổng khối lượng các đồ vật không quá M;
- Có không quá một vật có khối lượng chẵn được xếp vào giỏ;
- Số đồ vật được xếp là nhiều nhất.

Dữ liệu cho trong file DOVAT.INP gồm:

- Dòng đầu ghi hai số nguyên dương n và M $(n \le 10^5, M \le 10^9)$.
- Dòng thứ hai ghi n số nguyên dương $a_1, a_2, .., a_n$ $(a_i \le 10^6)$ là khối lượng của n đồ vật.

Kết quả ghi ra file DOVAT.OUT là số các đồ vật nhiều nhất có thể bỏ vào giỏ.

Ví dụ:

DOVAT.INP	DOVAT.OUT
4 18 1 6 4 7 9	3

Giải thích: Chọn các vật có khối lượng: 1, 4, 7.

2☼. Dãy răng cưa

Dãy số $a_1, a_2, \ldots, a_{2n+1}$ được gọi là dãy răng cưa nếu như $a_1 < a_2 > a_3 < a_4 > \ldots < a_{2n} > a_{2n+1}$.

Yêu cầu: Cho dãy các số nguyên $B = (b_1, b_2, \ldots, b_{2n+1})$, hãy kiểm tra xem có thể sắp xếp lại các phần tử của B sao cho dãy thu được là dãy răng cưa hay không?

Dữ liệu: Vào từ file văn bản PILSEQ.INP:

- Dòng đầu tiên chứa số nguyên k là số lượng dãy số cần kiểm tra;
- Tiếp đến là k nhóm dòng, mỗi nhóm gồm 2 dòng, chứa thông tin tương ứng với một dãy số cần kiểm tra: Dòng đầu tiên trong một nhóm dòng chứa số nguyên dương lẻ là số lượng phần tử trong dãy số, dòng thứ hai chứa các phần tử của dãy số được ghi cách nhau bởi dấu cách. Giả thiết: Số lượng số hạng của mỗi dãy số là không vượt quá 10^6 và mỗi số hạng có trị tuyệt đối không vượt quá 10^9.

Kết quả: Ghi ra k dòng của file văn bản PILSEQ.OUT: Dòng thứ i ghi '***YES***' nếu như có thể sắp xếp các phần tử của dãy thứ i trong file dữ liệu vào để thu được dãy răng cưa và ghi '***NO***' nếu trái lại.

Ví dụ:

PILSEQ.INP	PILSEQ.OUT
2 3 1 2 3 5 8 8 8 8 1	YES NO

3☼. Chọn tập – SelectSet.Cpp

Cho dãy số nguyên $A_1, A_2, .., A_N$ $(0 \le A_i \le 100)$ và số nguyên dương M $(M \le 100)$.

Yêu cầu: Hãy tìm cách chọn được nhiều số hạng nhất thuộc dãy sao cho tổng hai số bất kì được chọn **không chia hết cho *M***.

Dữ liệu cho trong file **SelectSet.Inp** gồm:

- Dòng đầu ghi hai số nguyên dương N $(N \le 100000)$ và M.
- Dòng thứ 2 ghi N số nguyên $A_1, A_2, .., A_N$.

Kết quả ghi ra file **SelectSet.Out** số các số hạng nhiều nhất có thể chọn được.

Ví dụ:

SelectSet.Inp	**SelectSet.Out**	Giải thích
6 5 1 2 3 4 4 4	4	Ta có thể chọn 5 số hạng: 2, 4, 4, 4.

4☼. Xếp lịch cuộc họp

Có n cuộc họp cần sử dụng phòng họp A. Các cuộc họp được đánh số thứ tự từ 1, 2, 3, .., n. Cuộc họp thứ i có thời gian bắt đầu là T_i và thời gian kết thúc là F_i . Hãy tìm cách xếp (chọn) được nhiều cuộc họp nhất. Hai cuộc họp có thể cùng được chọn nếu khoảng thời gian họp (*từ thời điểm bắt đầu đến thời điểm kết thúc*) không phủ chồng nhau. Thời điểm kết thúc của cuộc họp này có thể là thời điểm bắt đầu của cuộc họp khác.

Dữ liệu cho trong file SELECT.INP gồm:

- Dòng đầu ghi số nguyên n $(n \le 10^5)$.
- n dòng tiếp theo, mỗi dòng ghi hai số nguyên dương T_i và F_i là thời điểm bắt đầu và kết thúc của buổi họp thứ i $(T_i < F_i \le 10^9)$.

Kết quả ghi ra file SELECT.OUT là số các cuộc họp nhiều nhất có thể xếp để sử dụng phòng họp A.

Ví dụ:

SELECT.INP	SELECT.OUT
4 1 2 2 4 3 6 5 6	3

5☼. Những chiếc lều

Lấy cảm hứng từ những công trình kiến trúc nổi tiếng của kiến trúc sư Võ Trọng Nghĩa, Tèo cũng muốn thiết kế riêng cho mình một khu vườn thật đẹp mắt. Một trong nhũn công việc đó là thiết kế mái che cho hành lang đi vào. Có thể xem đoạn đường là một đoạn thẳng từ tọa độ x đến tọa độ y ($x < y$). Để che hành lang đi vào, Tèo sử dụng n chiếc lều, chiếc lều thứ i có thể che mưa được từ tọa độ x_i đến y_i ($x_i < y_i$) trên đoạn đường đi vào. Khi thiết kế xong, Tèo muốn biết n mái che này có che được mưa cho cả đoạn đường không? Bạn hãy lập trình giúp Tèo trả lời câu hỏi này nhé.

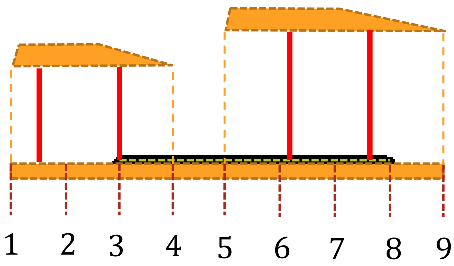
Dữ liệu cho trong file TENT.INP gồm:

- Dòng đầu ghi số T là số bộ dữ liệu kiểm tra ($T \le 10$).
- Theo sau là T nhóm dòng, mỗi nhóm dòng gồm:
- Dòng thứ nhất ghi ba số nguyên n, x, y tương ứng là số lều và tọa độ của đoạn đường.
- n dòng sau, mỗi dòng ghi hai số nguyên x_i, y_i là tọa độ có thể che mưa được của chiếc lều thứ i.

Kết quả ghi ra file TENT.OUT gồm T dòng tương ứng với T bộ dữ liệu. Nếu có thể che mưa được cả đoạn đường thì ghi 1, ngược lại ghi 0.

Ví dụ:

TENT.INP	TENT.OUT
2 2 3 8 1 4 5 9 3 2 8 2 4 3 7 5 10	0 1



Hình vẽ minh họa bộ dữ liệu thứ nhất.

Giới hạn:

- Sub1: $n \le 100$, $y - x \le 100$;
- Sub2: $n \le 10^5$, $|x|$, $|y|$, $|x_i|$, $|y_i| \le 10^9$.

6☼. Biến đổi đối xứng

Dãy số $A[1], A[2], ..., A[n]$ được gọi là dãy đối xứng nếu $A[i] = A[n-i+1]$ với $i = 1, 2, …, n$.

Cho dãy $A[1], A[2], ... , A[n]$. Ta thực hiện các phép biến đổi trên dãy như sau:

Chọn hai số hạng kề nhau, thay thế hai số hạng đó bằng tổng của chúng, ta được một dãy mới, ít hơn dãy trước khi biến đổi một số hạng.

Yêu cầu: Cần phải biến đổi dãy ít nhất bao nhiêu lần để biến đổi dãy $A[1], A[2], ..., A[n]$ thành dãy đối xứng.

Dữ liệu cho trong file **MPALIN.INP** như sau:

- Dòng đầu ghi số nguyên dương n;
- Dòng sau ghi n số nguyên dương $A[1], A[2], … , A[n]$. ($A[i] \le 10^9$).

Kết quả ghi ra file **MPALIN.OUT** là số lần biến đổi ít nhất để được dãy đối xứng.

Ví dụ:

MPALIN.INP	MPALIN.OUT	Giải thích
3 1 2 3	1	**1, 2,** 3→ 3, 3
5 1 2 4 6 1	1	1, **2, 4**, 6, 1→ 1, 6, 6, 1
4 1 4 3 2	2	**1, 4**, 3, 2 → 5, **3, 2**→ 5, 5

Giới hạn:

- 30% số test $n \le 10$;
- 60% số test $n \le 1000$;
- 10% số test $n \le 1000000$.

7☼. Biến đổi dãy số

Cho dãy số nguyên **X** gồm N số hạng $X_1, X_2, .. , X_N$. Ban đầu $X_1 = X_2 = ... = X_N = 0$.

Cho dãy số nguyên **A** gồm N số hạng $A_1, A_2, .. , A_N$.

Ta cần biến đổi dãy **X** thành dãy **A**, tức là $X_i = A_i$ với $i = 1, 2, 3, .., N$ bằng cách thực hiện:

Mỗi lần thực hiện: Chọn một chỉ số i ($1 \le i < N$), thay X_{i+1} thành $X_i + 1$.

Yêu cầu: Xác định xem, có thể biến đổi dãy **X** thành dãy **A** được hay không? Nếu biến đổi được, tìm số lần biến đổi ít nhất cần thực hiện.

Dữ liệu cho trong file RefSeq.Inp gồm:

- Dòng 1 ghi số nguyên dương N.
- Dòng 2 ghi N số nguyên $A_1, A_2, .. , A_N$.

Kết quả ghi ra file RefSeq.Out là số lần biến đổi ít nhất cần thực hiện để đưa dãy **X** thành dãy **A**. Nếu không có cách để biến đổi thì ghi -1.

Ví dụ:

RefSeq.Inp	RefSeq.Out
4 0 1 1 2	3
3	-1

1 2 1	

Giải thích:

Ban đầu: **X** = (0, 0, 0, 0);

Lần 1: Chọn chỉ số 3: $X_3 = X_2 + 1 \rightarrow$ **X** = (0, 0, 1, 0);

Lần 2: Chọn chỉ số 2: $X_2 = X_1 + 1 \rightarrow$ **X** = (0, 1, 1, 0);

Lần 3: Chọn chỉ số 4: $X_4 = X_3 + 1 \rightarrow$ **X** = (0, 1, 1, 2) = **A**.

Giới hạn: $N \leq 10^5$; $0 \leq A_i \leq 10^9$;