Thuật toán ứng dụng
Bài thực hành số 5.1: Tham lam

TS. Bùi Quốc Trung, TA. Đặng Xuân Vương

Trường Đại học Bách khoa Hà Nội
Viện Công nghệ thông tin và Truyền thông

Ngày 31 tháng 5 năm 2021

Mục lục

Mục lục

MỞ ĐẦU

- Tham lam là một mô hình giải thuật.

MỞ ĐẦU

- Tham lam là một mô hình giải thuật.
- Thuật toán tham lam tìm cách đưa ra lựa chọn tối ưu cục bộ tại mỗi bước với hy vọng sẽ tìm được một tối ưu toàn cục.

MỞ ĐẦU

- Tham lam là một mô hình giải thuật.
- Thuật toán tham lam tìm cách đưa ra lựa chọn tối ưu cục bộ tại mỗi bước với hy vọng sẽ tìm được một tối ưu toàn cục.
- Tham lam là một thuật toán heuristic.

MỞ ĐẦU

- Tham lam là một mô hình giải thuật.
- Thuật toán tham lam tìm cách đưa ra lựa chọn tối ưu cục bộ tại mỗi bước với hy vọng sẽ tìm được một tối ưu toàn cục.
- Tham lam là một thuật toán heuristic.
- Thuật toán tham lam thường không đưa ra lời giải tối ưu.

MỞ ĐẦU

- Tham lam là một mô hình giải thuật.
- Thuật toán tham lam tìm cách đưa ra lựa chọn tối ưu cục bộ tại mỗi bước với hy vọng sẽ tìm được một tối ưu toàn cục.
- Tham lam là một thuật toán heuristic.
- Thuật toán tham lam thường không đưa ra lời giải tối ưu.
- Được sử dụng khi không tìm được thuật toán chính xác chạy trong thời gian chấp nhận được.

Mục lục

CHANGE

- Có các đồng tiền mệnh giá lần lượt là \$1, \$5, \$10, \$50, \$100, \$500.

CHANGE

- Có các đồng tiền mệnh giá lần lượt là \$1, \$5, \$10, \$50, \$100, \$500.
- Sử dụng các đồng tiền có mệnh giá như trên để tạo ra tổng tiền \$N, sao cho số đồng tiền được sử dụng là ít nhất.

CHANGE

- Có các đồng tiền mệnh giá lần lượt là \$1, \$5, \$10, \$50, \$100, \$500.
- Sử dụng các đồng tiền có mệnh giá như trên để tạo ra tổng tiền \$N, sao cho số đồng tiền được sử dụng là ít nhất.
- Giới hạn: $1 \leq N \leq 999$.

Thuật toán

- **Thuật toán 1 - Vét cạn:** Duyệt vét cạn tất cả các cách chia tiền và chọn cách chia sử dụng ít đồng tiền nhất.

Thuật toán

- **Thuật toán 1 - Vét cạn:** Duyệt vét cạn tất cả các cách chia tiền và chọn cách chia sử dụng ít đồng tiền nhất.
- **Thuật toán 2 - Chia để trị**: Với $n \in \{1, 5, 10, 50, 100, 500\}$ thì $F[n] = 1$, ngược lại $F[n] = min_{i \in [1, n-1]} F[i] + F[n-i]$.

Thuật toán

- **Thuật toán 1 - Vét cạn:** Duyệt vét cạn tất cả các cách chia tiền và chọn cách chia sử dụng ít đồng tiền nhất.
- **Thuật toán 2 - Chia để trị**: Với $n \in \{1, 5, 10, 50, 100, 500\}$ thì $F[n] = 1$, ngược lại $F[n] = min_{i \in [1, n-1]} F[i] + F[n-i]$.
- **Thuật toán 3 - Quy hoạch động**: Với $n \in \{1, 5, 10, 50, 100, 500\}$ thì $F[n] = 1$, $n = 0$ thì $F[n] = 0$, $n < 0$ thì $F[n] = \infty$, ngược lại $F[n] = min(F[n-i] + 1), \forall i \in \{1, 5, 10, 50, 100, 500\}$.

Thuật toán

- **Thuật toán 1 - Vét cạn:** Duyệt vét cạn tất cả các cách chia tiền và chọn cách chia sử dụng ít đồng tiền nhất.
- **Thuật toán 2 - Chia để trị**: Với $n \in \{1, 5, 10, 50, 100, 500\}$ thì $F[n] = 1$, ngược lại $F[n] = min_{i \in [1, n-1]} F[i] + F[n-i]$.
- **Thuật toán 3 - Quy hoạch động**: Với $n \in \{1, 5, 10, 50, 100, 500\}$ thì $F[n] = 1$, $n = 0$ thì $F[n] = 0$, $n < 0$ thì $F[n] = \infty$, ngược lại $F[n] = min(F[n-i] + 1), \forall i \in \{1, 5, 10, 50, 100, 500\}$.
- **Thuật toán 4 - Tham lam:** Xét lần lượt các đồng tiền có mệnh giá từ cao đến thấp, với mỗi mệnh giá, chọn số đồng tiền lớn nhất sao cho tổng giá trị của các đồng tiền được chọn không vượt quá $N.

Chứng minh

- Thuật toán tham lam luôn đưa ra một lời giải chấp nhận được (chọn ra được các đồng tiền có tổng mệnh giá bằng $N):

Chứng minh

- Thuật toán tham lam luôn đưa ra một lời giải chấp nhận được (chọn ra được các đồng tiền có tổng mệnh giá bằng $N):
  - Tồn tại đồng tiền mệnh giá $1.

Chứng minh

- Thuật toán tham lam luôn đưa ra một lời giải chấp nhận được (chọn ra được các đồng tiền có tổng mệnh giá bằng \$N):
  - Tồn tại đồng tiền mệnh giá \$1.
  - N và mệnh giá của tất cả các đồng tiền đều chia hết cho 1.

Chứng minh

- Luôn tồn tại một lời giải tối ưu mà với mọi đồng tiền có mệnh giá \$X, tổng giá trị của các đồng tiền có mệnh giá nhỏ hơn \$X được chọn nhỏ hơn \$X:

Chứng minh

- Luôn tồn tại một lời giải tối ưu mà với mọi đồng tiền có mệnh giá \$X, tổng giá trị của các đồng tiền có mệnh giá nhỏ hơn \$X được chọn nhỏ hơn \$X:
  - Số đồng tiền \$1 được chọn nhỏ hơn 5.

Chứng minh

- Luôn tồn tại một lời giải tối ưu mà với mọi đồng tiền có mệnh giá \$X, tổng giá trị của các đồng tiền có mệnh giá nhỏ hơn \$X được chọn nhỏ hơn \$X:
  - Số đồng tiền \$1 được chọn nhỏ hơn 5.
  - Số đồng tiền \$5 được chọn nhỏ hơn 2.

Chứng minh

- Luôn tồn tại một lời giải tối ưu mà với mọi đồng tiền có mệnh giá \$X, tổng giá trị của các đồng tiền có mệnh giá nhỏ hơn \$X được chọn nhỏ hơn \$X:
  - Số đồng tiền \$1 được chọn nhỏ hơn 5.
  - Số đồng tiền \$5 được chọn nhỏ hơn 2.
  - Từ 2 điều trên, suy ra tổng giá trị của các đồng tiền mệnh giá \$1 và \$5 nhỏ hơn \$10.

Chứng minh

- Luôn tồn tại một lời giải tối ưu mà với mọi đồng tiền có mệnh giá \$X, tổng giá trị của các đồng tiền có mệnh giá nhỏ hơn \$X được chọn nhỏ hơn \$X:
  - Số đồng tiền \$1 được chọn nhỏ hơn 5.
  - Số đồng tiền \$5 được chọn nhỏ hơn 2.
  - Từ 2 điều trên, suy ra tổng giá trị của các đồng tiền mệnh giá \$1 và \$5 nhỏ hơn \$10.
  - Tương tự với các mệnh giá lớn hơn.

Chứng minh

- Luôn tồn tại một lời giải tối ưu mà với mọi đồng tiền có mệnh giá \$X, tổng giá trị của các đồng tiền có mệnh giá nhỏ hơn \$X được chọn nhỏ hơn \$X:
  - Số đồng tiền \$1 được chọn nhỏ hơn 5.
  - Số đồng tiền \$5 được chọn nhỏ hơn 2.
  - Từ 2 điều trên, suy ra tổng giá trị của các đồng tiền mệnh giá \$1 và \$5 nhỏ hơn \$10.
  - Tương tự với các mệnh giá lớn hơn.
- Nói cách khác, từ một lời giải chấp nhận được bất kỳ, ta có thể biến đổi thành một lời giải không tồi hơn, thoả mãn tính chất trên.

Chứng minh

- Giả sử tồn tại một lời giải chấp nhận được dùng đúng a đồng \$500, b là tổng giá trị các đồng tiền có mệnh giá nhỏ hơn \$500.

Chứng minh

- Giả sử tồn tại một lời giải chấp nhận được dùng đúng a đồng \$500, b là tổng giá trị các đồng tiền có mệnh giá nhỏ hơn \$500.
- Nếu $b \geq 500$, ta có thể biến đổi lời giải hiện tại thành lời giải sử dụng $a + 1$ đồng \$500 và tổng giá trị các đồng tiền có mệnh giá nhỏ hơn \$500 là $b - 500$.

Chứng minh

- Giả sử tồn tại một lời giải chấp nhận được dùng đúng a đồng \$500, b là tổng giá trị các đồng tiền có mệnh giá nhỏ hơn \$500.
- Nếu $b \geq 500$, ta có thể biến đổi lời giải hiện tại thành lời giải sử dụng $a+1$ đồng \$500 và tổng giá trị các đồng tiền có mệnh giá nhỏ hơn \$500 là $b-500$.
- Thực hiện việc biến đổi cho đến khi không thể tăng a được nữa.

Chứng minh

- Giả sử tồn tại một lời giải chấp nhận được dùng đúng a đồng \$500, b là tổng giá trị các đồng tiền có mệnh giá nhỏ hơn \$500.
- Nếu $b \geq 500$, ta có thể biến đổi lời giải hiện tại thành lời giải sử dụng $a + 1$ đồng \$500 và tổng giá trị các đồng tiền có mệnh giá nhỏ hơn \$500 là $b - 500$.
- Thực hiện việc biến đổi cho đến khi không thể tăng a được nữa.
- Tương tự với các mệnh giá khác, xét theo thứ tự giảm dần.

Code

```
int a[6] = {1, 5, 10, 50, 100, 500};
int res = 0;
for (int i = 5; i >= 0; i--) {
    res += n / a[i];
    n %= a[i];
}
cout << res << endl;
```

Mở rộng

- Thuật toán tham lam trên vẫn đúng khi các tờ tiền có mệnh giá $\$10^i$ và $\$5 \times 10^i$, $\forall i \leq c$, c lớn bất kỳ.
- Từ chứng minh trên, ta cũng có ý tưởng về một thuật toán (hoặc cách cài đặt khác):
    - Khởi tạo: $i = 1$.
    - Xét phần lẻ trong số tiền còn phải trả: $T = N\%10^i$.
    - Đếm số tờ tiền tối thiểu có tổng bằng T và cộng vào kết quả.
    - Cập nhật số tiền còn phải trả $N = N - T$ và cập nhật $i = i + 1$.
    - Lặp lại từ bước 2 cho đến khi $N = 0$.

Code

```
int c[10] = {0, 1, 2, 3, 4, 1, 2, 3, 4, 5};
int res = 0;
int x = 10; // x = 10^i;
while (N > 0) {
    int T = N % x;
    if (T >= 1000) res += T / 500;
    else res += c[T*10/x];
    x *= 10;
    N -= T;
}
cout << res << endl;
```

Mục lục

ATM

- Các đồng tiền có mệnh giá là 1000×10^i, 2000×10^i, 3000×10^i, 5000×10^i với $i \le c$.

ATM

- Các đồng tiền có mệnh giá là 1000×10^i, 2000×10^i, 3000×10^i, 5000×10^i với $i \leq c$.
- Sử dụng các đồng tiền có mệnh giá như trên để tạo ra tổng tiền \$W, sao cho số đồng tiền được sử dụng là ít nhất.

ATM

- Các đồng tiền có mệnh giá là 1000×10^i, 2000×10^i, 3000×10^i, 5000×10^i với $i \leq c$.
- Sử dụng các đồng tiền có mệnh giá như trên để tạo ra tổng tiền \$W, sao cho số đồng tiền được sử dụng là ít nhất.
- Giới hạn: $1 \leq W \leq 10^{18}$, $c \leq 15$.

Thuật toán

- $W\%1000 = 0$: Luôn có lời giải.
- $W\%1000 \neq 0$: Không có lời giải.
- Hệ quả: Ta có thể chia W và các mệnh giá tiền cho 1000 trong trường hợp có lời giải.

Thuật toán

- Chứng minh tương tự bài CHANGE, trong lời giải tối ưu, tổng giá trị các tờ tiền có mệnh giá nhỏ hơn 10^i **luôn luôn** nhỏ hơn 10^i, $i \leq c$.
  - Có tối đa 1 tờ \$1x.
  - Có tối đa 2 tờ \$2x.
  - Tổng giá trị của những tờ \$1x và \$2x nhỏ hơn \$5x.
  - Có tối đa 3 tờ \$3x.
  - Không thể có đồng thời tờ \$2x và tờ \$3x.
  - Không thể có đồng thời tờ \$1x và 2 tờ \$3x.
  - Có tối đa 1 tờ \$5x.
  - Không thể có đồng thời tờ \$5x và 2 tờ \$3x.

Thuật toán

- Chứng minh tương tự bài CHANGE, trong lời giải tối ưu, tổng giá trị các tờ tiền có mệnh giá nhỏ hơn 10^i **luôn luôn** nhỏ hơn 10^i, $i \leq c$.
  - Có tối đa 1 tờ \$1x.
  - Có tối đa 2 tờ \$2x.
  - Tổng giá trị của những tờ \$1x và \$2x nhỏ hơn \$5x.
  - Có tối đa 3 tờ \$3x.
  - Không thể có đồng thời tờ \$2x và tờ \$3x.
  - Không thể có đồng thời tờ \$1x và 2 tờ \$3x.
  - Có tối đa 1 tờ \$5x.
  - Không thể có đồng thời tờ \$5x và 2 tờ \$3x.
- Ta có thể cài đặt tương tự phần mở rộng của bài CHANGE.

Thuật toán

- Chứng minh tương tự bài CHANGE, trong lời giải tối ưu, tổng giá trị các tờ tiền có mệnh giá nhỏ hơn 10^i **luôn luôn** nhỏ hơn 10^i, $i \leq c$.
  - Có tối đa 1 tờ $1x.
  - Có tối đa 2 tờ $2x.
  - Tổng giá trị của những tờ $1x và $2x nhỏ hơn $5x.
  - Có tối đa 3 tờ $3x.
  - Không thể có đồng thời tờ $2x và tờ $3x.
  - Không thể có đồng thời tờ $1x và 2 tờ $3x.
  - Có tối đa 1 tờ $5x.
  - Không thể có đồng thời tờ $5x và 2 tờ $3x.
- Ta có thể cài đặt tương tự phần mở rộng của bài CHANGE.
  - Xét lần lượt từng phần lẻ của W.

Thuật toán

- Chứng minh tương tự bài CHANGE, trong lời giải tối ưu, tổng giá trị các tờ tiền có mệnh giá nhỏ hơn 10^i **luôn luôn** nhỏ hơn 10^i, $i \leq c$.
  - Có tối đa 1 tờ \$1x.
  - Có tối đa 2 tờ \$2x.
  - Tổng giá trị của những tờ \$1x và \$2x nhỏ hơn \$5x.
  - Có tối đa 3 tờ \$3x.
  - Không thể có đồng thời tờ \$2x và tờ \$3x.
  - Không thể có đồng thời tờ \$1x và 2 tờ \$3x.
  - Có tối đa 1 tờ \$5x.
  - Không thể có đồng thời tờ \$5x và 2 tờ \$3x.
- Ta có thể cài đặt tương tự phần mở rộng của bài CHANGE.
  - Xét lần lượt từng phần lẻ của W.
  - Do tổng giá trị các tờ $t \times 10^c, t \in \{1000, 2000, 3000\}$ có thể lớn hơn 5000×10^c nên cần xét riêng trường hợp này.

Thuật toán

- w: Giá trị phần lẻ
- N_c: Số tờ tiền tối thiểu
- N_s: Số lời giải tối ưu

w	0	1	2	3	4	5	6	7	8	9
N_c	0	1	1	1	2	1	2	2	2	3
N_s	1	1	1	1	2	1	2	1	1	3

Code

```
w /= 1000;
ll ns = 1;
ll nc = 0;
for (int i = 0; i < c; i++) {
    t = w % 10;
    nc += Nc[t];
    ns *= Ns[t];
    w /= 10;
}
```

Thuật toán

Sau khi xử lý xong phần lẻ, $W\%10^c = 0$, đặt $M = W/10^c$, bài toán trở thành tìm cách tạo ra tổng $Mx từ các tờ tiền $1x, $2x, $3x và $5x:

- $M < 5$: Sử dụng mảng hằng như thuật toán phía trên.

Thuật toán

Sau khi xử lý xong phần lẻ, $W\%10^c = 0$, đặt $M = W/10^c$, bài toán trở thành tìm cách tạo ra tổng \$Mx từ các tờ tiền \$1x, \$2x, \$3x và \$5x:

- $M < 5$: Sử dụng mảng hằng như thuật toán phía trên.
- $M \geq 5$: Ta vẫn dễ dàng chứng minh được **tồn tại ít nhất một** lời giải tối ưu mà tổng giá trị các tờ \$1x, \$2x, \$3x nhỏ hơn \$5x.
    - 2 tờ \$3x được thay thành 1 tờ \$5x và 1 tờ \$1x.

Thuật toán

- Ta tìm ra được số tờ tiền tối thiểu bằng thuật toán tham lam (tương tự thuật toán ban đầu của bài CHANGE):

Thuật toán

- Ta tìm ra được số tờ tiền tối thiểu bằng thuật toán tham lam (tương tự thuật toán ban đầu của bài CHANGE):
  - Lời giải là các tờ \$5x và các tờ \$1x, \$2x, \$3x có tổng mệnh giá nhỏ hơn \$5x (gọi tổng này là x).

Thuật toán

- Ta tìm ra được số tờ tiền tối thiểu bằng thuật toán tham lam (tương tự thuật toán ban đầu của bài CHANGE):
  - Lời giải là các tờ \$5x và các tờ \$1x, \$2x, \$3x có tổng mệnh giá nhỏ hơn \$5x (gọi tổng này là x).
  - Xem xét từng trường hợp của x, ta cố gắng tìm lời giải thay thế bằng cách kết hợp một số đồng \$5x với các đồng mệnh giá nhỏ hơn và tạo ra các đồng mới sao cho số lượng và tổng giá trị đều không đổi:

Thuật toán

- Ta tìm ra được số tờ tiền tối thiểu bằng thuật toán tham lam (tương tự thuật toán ban đầu của bài CHANGE):
  - Lời giải là các tờ \$5x và các tờ \$1x, \$2x, \$3x có tổng mệnh giá nhỏ hơn \$5x (gọi tổng này là x).
  - Xem xét từng trường hợp của x, ta cố gắng tìm lời giải thay thế bằng cách kết hợp một số đồng \$5x với các đồng mệnh giá nhỏ hơn và tạo ra các đồng mới sao cho số lượng và tổng giá trị đều không đổi:
    - $x = 0$: Không có lời giải thay thế.

Thuật toán

- Ta tìm ra được số tờ tiền tối thiểu bằng thuật toán tham lam (tương tự thuật toán ban đầu của bài CHANGE):
  - Lời giải là các tờ \$5x và các tờ \$1x, \$2x, \$3x có tổng mệnh giá nhỏ hơn \$5x (gọi tổng này là x).
  - Xem xét từng trường hợp của x, ta cố gắng tìm lời giải thay thế bằng cách kết hợp một số đồng \$5x với các đồng mệnh giá nhỏ hơn và tạo ra các đồng mới sao cho số lượng và tổng giá trị đều không đổi:
    - $x = 0$: Không có lời giải thay thế.
    - $x = 1$: Có 1 lời giải thay thế 1 tờ \$5. Không thể có lời giải thay thế nhiều hơn 1 tờ \$5.

Thuật toán

- Ta tìm ra được số tờ tiền tối thiểu bằng thuật toán tham lam (tương tự thuật toán ban đầu của bài CHANGE):
  - Lời giải là các tờ \$5x và các tờ \$1x, \$2x, \$3x có tổng mệnh giá nhỏ hơn \$5x (gọi tổng này là x).
  - Xem xét từng trường hợp của x, ta cố gắng tìm lời giải thay thế bằng cách kết hợp một số đồng \$5x với các đồng mệnh giá nhỏ hơn và tạo ra các đồng mới sao cho số lượng và tổng giá trị đều không đổi:
    - $x = 0$: Không có lời giải thay thế.
    - $x = 1$: Có 1 lời giải thay thế 1 tờ \$5. Không thể có lời giải thay thế nhiều hơn 1 tờ \$5.
    - $x = 2$: Không có lời giải thay thế.

Thuật toán

- Ta tìm ra được số tờ tiền tối thiểu bằng thuật toán tham lam (tương tự thuật toán ban đầu của bài CHANGE):
  - Lời giải là các tờ \$5x và các tờ \$1x, \$2x, \$3x có tổng mệnh giá nhỏ hơn \$5x (gọi tổng này là x).
  - Xem xét từng trường hợp của x, ta cố gắng tìm lời giải thay thế bằng cách kết hợp một số đồng \$5x với các đồng mệnh giá nhỏ hơn và tạo ra các đồng mới sao cho số lượng và tổng giá trị đều không đổi:
    - $x = 0$: Không có lời giải thay thế.
    - $x = 1$: Có 1 lời giải thay thế 1 tờ \$5. Không thể có lời giải thay thế nhiều hơn 1 tờ \$5.
    - $x = 2$: Không có lời giải thay thế.
    - $x = 3$: Không có lời giải thay thế.

Thuật toán

- Ta tìm ra được số tờ tiền tối thiểu bằng thuật toán tham lam (tương tự thuật toán ban đầu của bài CHANGE):
  - Lời giải là các tờ \$5x và các tờ \$1x, \$2x, \$3x có tổng mệnh giá nhỏ hơn \$5x (gọi tổng này là x).
  - Xem xét từng trường hợp của x, ta cố gắng tìm lời giải thay thế bằng cách kết hợp một số đồng \$5x với các đồng mệnh giá nhỏ hơn và tạo ra các đồng mới sao cho số lượng và tổng giá trị đều không đổi:
    - $x = 0$: Không có lời giải thay thế.
    - $x = 1$: Có 1 lời giải thay thế 1 tờ \$5. Không thể có lời giải thay thế nhiều hơn 1 tờ \$5.
    - $x = 2$: Không có lời giải thay thế.
    - $x = 3$: Không có lời giải thay thế.
    - $x = 4$: Khi không thay thế tờ \$5 nào, trường hợp này vốn có 2 lời giải. Có 1 lời giải thay thế 1 tờ \$5. Không thể có lời giải thay thế nhiều hơn 1 tờ \$5.

Code

```
t = w % 5;
nc += w / 5 + Nc[t];
int a = w >= 5 ? 1 : 0;
if (t == 1) ns *= (1 + a);
else if (t == 4) ns *= (2 + a);
```

Code

```
while (n_test--) {
    cin >> w >> c;
    if (w % 1000) {
        cout << 0 << endl;
        continue;
    }
    w /= 1000;
    ll ns = 1;
    ll nc = 0;
    // Xu ly phan le
    // Truong hop dac biet
    cout << nc << ' ' << ns << endl;
}
```

Mục lục

PLANTING TREES

- Có N hạt giống.
- Hạt giống i cần thời gian t_i để phát triển thành cây.
- Mỗi ngày chỉ được gieo 1 hạt giống.
- **Yêu cầu:** Ngày sớm nhất tất cả hạt giống phát triển thành cây.
- **Giới hạn:**
  - $N \leq 10^5$.
  - $t_i \leq 10^6$.

Nhận xét

- Nếu hạt giống i được trồng vào ngày x có thời gian phát triển nhỏ hơn hạt giống j được trồng vào ngày $x + 1$.
  - Thời gian sớm nhất để 2 cây cùng phát triển là $t_j + x + 1$.
  - Nếu đổi thời gian trồng của 2 hạt giống, thời gian sớm nhất để 2 cây cùng phát triển là $max(t_i + x + 1, j + x) < j + x + 1$.
  - Nếu đổi chỗ tất cả các cặp hạt i, j như vậy, ta thu được dãy sắp xếp giảm dần của thời gian phát triển.

Thuật toán

- Sắp xếp lại các hạt giống theo thứ tự giảm dần của thời gian phát triển.
- Gieo các hạt giống theo thứ tự đã sắp xếp.

Chứng minh

- Lời giải của bài toán về bản chất là thứ tự các hạt được trồng.

Chứng minh

- Lời giải của bài toán về bản chất là thứ tự các hạt được trồng.
- Từ lời giải thu được từ thuật toán tham lam, ta chọn 1 tập hạt giống và hoán đổi vị trí của chúng:

Chứng minh

- Lời giải của bài toán về bản chất là thứ tự các hạt được trồng.
- Từ lời giải thu được từ thuật toán tham lam, ta chọn 1 tập hạt giống và hoán đổi vị trí của chúng:
  - Bằng cách này, toàn bộ không gian lời giải đều được sinh ra.

Chứng minh

- Lời giải của bài toán về bản chất là thứ tự các hạt được trồng.
- Từ lời giải thu được từ thuật toán tham lam, ta chọn 1 tập hạt giống và hoán đổi vị trí của chúng:
  - Bằng cách này, toàn bộ không gian lời giải đều được sinh ra.
  - Không lời giải nào được sinh ra tốt hơn lời giải ban đầu.

Chứng minh

- Lời giải của bài toán về bản chất là thứ tự các hạt được trồng.
- Từ lời giải thu được từ thuật toán tham lam, ta chọn 1 tập hạt giống và hoán đổi vị trí của chúng:
  - Bằng cách này, toàn bộ không gian lời giải đều được sinh ra.
  - Không lời giải nào được sinh ra tốt hơn lời giải ban đầu.
  - Giả sử hạt giống được trồng vào ngày x là i, $t_i + x$ lớn nhất trong các hạt được chọn. Sau khi hoán đổi:

Chứng minh

- Lời giải của bài toán về bản chất là thứ tự các hạt được trồng.
- Từ lời giải thu được từ thuật toán tham lam, ta chọn 1 tập hạt giống và hoán đổi vị trí của chúng:
  - Bằng cách này, toàn bộ không gian lời giải đều được sinh ra.
  - Không lời giải nào được sinh ra tốt hơn lời giải ban đầu.
  - Giả sử hạt giống được trồng vào ngày x là i, $t_i + x$ lớn nhất trong các hạt được chọn. Sau khi hoán đổi:
    - i được trồng ở ngày $y \geq x$.

Chứng minh

- Lời giải của bài toán về bản chất là thứ tự các hạt được trồng.
- Từ lời giải thu được từ thuật toán tham lam, ta chọn 1 tập hạt giống và hoán đổi vị trí của chúng:
  - Bằng cách này, toàn bộ không gian lời giải đều được sinh ra.
  - Không lời giải nào được sinh ra tốt hơn lời giải ban đầu.
  - Giả sử hạt giống được trồng vào ngày x là i, $t_i + x$ lớn nhất trong các hạt được chọn. Sau khi hoán đổi:
    - i được trồng ở ngày $y \geq x$.
    - i được trồng ở ngày $y < x$, vậy phải có hạt j trước đó được trồng ở ngày $y_0 < x$ được trồng vào ngày $z \geq x$, mà $t_j \geq t_i$.

Code

```
sort(a.rbegin(), a.rend());
int res = 0;
for (int i = 0; i < n; i++) {
    res = max(res, i + a[i]);
}
cout << res + 2 << endl;
```

Mục lục

POSTMAN

- Có n khách hàng.
- Khách hàng i ở tọa độ x_i và cần được giao m_i kiện hàng.
- Mỗi chuyến đi chở được tối đa k kiện hàng.
- Kho ở tọa độ 0, thời gian di chuyển phụ thuộc khoảng cách, thời gian tháo dỡ hàng không đáng kể.
- **Yêu cầu:** Tìm thời gian tối thiểu để phục vụ n khách hàng.
- **Giới hạn:**
  - $n \leq 10^3$.
  - $k \leq 10^7$.
  - $|x_i| \leq 10^7$.
  - $m_i \leq 10^7$.

Thuật toán

- Chia khách hàng thành 2 nhóm: tọa độ < 0 và tọa độ > 0. Xét riêng từng nhóm.

Thuật toán

- Chia khách hàng thành 2 nhóm: tọa độ < 0 và tọa độ > 0. Xét riêng từng nhóm.
- Với mỗi nhóm, sắp xếp lại khách hàng theo khoảng cách đến kho (giảm dần):

Thuật toán

- Chia khách hàng thành 2 nhóm: tọa độ < 0 và tọa độ > 0. Xét riêng từng nhóm.
- Với mỗi nhóm, sắp xếp lại khách hàng theo khoảng cách đến kho (giảm dần):
  - Mỗi lần rời kho luôn mang theo k kiện hàng.

Thuật toán

- Chia khách hàng thành 2 nhóm: tọa độ < 0 và tọa độ > 0. Xét riêng từng nhóm.
- Với mỗi nhóm, sắp xếp lại khách hàng theo khoảng cách đến kho (giảm dần):
  - Mỗi lần rời kho luôn mang theo k kiện hàng.
  - Gọi i là chỉ số khách hàng ở xa nhất (sau khi sắp xếp) vẫn chưa được giao đủ hàng.

Thuật toán

- Chia khách hàng thành 2 nhóm: tọa độ < 0 và tọa độ > 0. Xét riêng từng nhóm.
- Với mỗi nhóm, sắp xếp lại khách hàng theo khoảng cách đến kho (giảm dần):
    - Mỗi lần rời kho luôn mang theo k kiện hàng.
    - Gọi i là chỉ số khách hàng ở xa nhất (sau khi sắp xếp) vẫn chưa được giao đủ hàng.
    - Ưu tiên giao các kiện hàng cho khách hàng i trước, nếu còn dư thì giao cho khách hàng $i + 1$, nếu vẫn còn dư thì tiếp tục giao cho khách hàng $i + 2, i + 3, \ldots$

Chứng minh

- Việc chia khách hàng thành 2 nhóm không ảnh hưởng đến lời giải bài toán:

Chứng minh

- Việc chia khách hàng thành 2 nhóm không ảnh hưởng đến lời giải bài toán:
  - 1 chuyến giao hàng cho khách hàng ở 2 nhóm tương đương với 2 chuyến đi, mỗi chuyến giao hàng cho khách hàng ở 1 nhóm.

Chứng minh

- Việc chia khách hàng thành 2 nhóm không ảnh hưởng đến lời giải bài toán:
  - 1 chuyến giao hàng cho khách hàng ở 2 nhóm tương đương với 2 chuyến đi, mỗi chuyến giao hàng cho khách hàng ở 1 nhóm.
  - Tiếp theo, ta chỉ xem xét bài toán trên 1 nhóm.

Chứng minh

- Việc chia khách hàng thành 2 nhóm không ảnh hưởng đến lời giải bài toán:
  - 1 chuyến giao hàng cho khách hàng ở 2 nhóm tương đương với 2 chuyến đi, mỗi chuyến giao hàng cho khách hàng ở 1 nhóm.
  - Tiếp theo, ta chỉ xem xét bài toán trên 1 nhóm.
- Mỗi chuyến đi đều cần giao đúng k kiện hàng (trừ chuyến cuối):

Chứng minh

- Việc chia khách hàng thành 2 nhóm không ảnh hưởng đến lời giải bài toán:
  - 1 chuyến giao hàng cho khách hàng ở 2 nhóm tương đương với 2 chuyến đi, mỗi chuyến giao hàng cho khách hàng ở 1 nhóm.
  - Tiếp theo, ta chỉ xem xét bài toán trên 1 nhóm.
- Mỗi chuyến đi đều cần giao đúng k kiện hàng (trừ chuyến cuối):
  - Nếu chuyến hàng t_1 đi xa hơn chuyến hàng t_2, ta có thể chuyển bớt hàng của chuyến t_2 sang chuyến t_1, kết quả giao hàng không đổi và tổng thời gian giao hàng không tăng.

Chứng minh

- Việc chia khách hàng thành 2 nhóm không ảnh hưởng đến lời giải bài toán:
  - 1 chuyến giao hàng cho khách hàng ở 2 nhóm tương đương với 2 chuyến đi, mỗi chuyến giao hàng cho khách hàng ở 1 nhóm.
  - Tiếp theo, ta chỉ xem xét bài toán trên 1 nhóm.
- Mỗi chuyến đi đều cần giao đúng k kiện hàng (trừ chuyến cuối):
  - Nếu chuyến hàng t_1 đi xa hơn chuyến hàng t_2, ta có thể chuyển bớt hàng của chuyến t_2 sang chuyến t_1, kết quả giao hàng không đổi và tổng thời gian giao hàng không tăng.
- Sau khi tất cả khách hàng trước i được giao đủ hàng (lượng hàng dư cũng đã được phân bổ cho các khách hàng còn lại), gọi m_j là số kiện hàng còn phải giao cho khách thứ j:
  - Ta cần dùng đúng $\lceil \frac{m_i}{k} \rceil$ chuyến giao hàng đến khách hàng i (MĐ1).
  - Lượng hàng dư được phân bổ ưu tiên cho khách hàng ở xa nhất luôn là lựa chọn tối ưu (MĐ2).

Chứng minh

Mệnh đề 1:

- Nếu dùng ít hơn $\lceil \frac{m_i}{k} \rceil$ chuyến, không thể giao đủ số hàng.

Chứng minh

Mệnh đề 1:

- Nếu dùng ít hơn $\lceil \frac{m_i}{k} \rceil$ chuyến, không thể giao đủ số hàng.
- Nếu dùng nhiều hơn $\lceil \frac{m_i}{k} \rceil$ chuyến:

Chứng minh

Mệnh đề 1:

- Nếu dùng ít hơn $\lceil \frac{m_i}{k} \rceil$ chuyến, không thể giao đủ số hàng.
- Nếu dùng nhiều hơn $\lceil \frac{m_i}{k} \rceil$ chuyến:
    - Gọi lượng hàng được giao cho khách thứ j là h_j, $h_i = m_i$.

Chứng minh

Mệnh đề 1:

- Nếu dùng ít hơn $\lceil \frac{m_i}{k} \rceil$ chuyến, không thể giao đủ số hàng.
- Nếu dùng nhiều hơn $\lceil \frac{m_i}{k} \rceil$ chuyến:
  - Gọi lượng hàng được giao cho khách thứ j là h_j, $h_i = m_i$.
  - Thay mỗi chuyến vượt quá thành một chuyến giao hàng chỉ đi tối đa đến $i + 1$ sẽ cho kết quả giao hàng tương tự, không làm ảnh hưởng đến các chuyến giao hàng khác nhưng tốn ít thời gian hơn.

Chứng minh

Mệnh đề 2:

- Sau khi khách hàng i nhận đủ hàng và còn một lượng hàng dư k':

Chứng minh

Mệnh đề 2:

- Sau khi khách hàng i nhận đủ hàng và còn một lượng hàng dư k':
  - $m_{i+1}\%k \neq 0, k' > m_{i+1}\%k$: Phân bổ $m_{i+1}\%k$ cho $i+1$ giúp giảm 1 chuyến giao hàng đến $i+1$, ta thêm 1 chuyến "bù"cho các khách hàng còn lại.

Chứng minh

Mệnh đề 2:

- Sau khi khách hàng i nhận đủ hàng và còn một lượng hàng dư k':
  - $m_{i+1}\%k \neq 0, k' > m_{i+1}\%k$: Phân bổ $m_{i+1}\%k$ cho $i+1$ giúp giảm 1 chuyến giao hàng đến $i+1$, ta thêm 1 chuyến "bù"cho các khách hàng còn lại.
  - Sau khi xem xét trường hợp trên, vẫn còn dư k'' kiện hàng: nếu phân bổ hết cho $i+1$, chuyến xe giao hàng cuối cùng đến $i+1$ dư ít nhất k'' kiện hàng.

Code

```
ll cal(vector<pll> x) {
    ll res = 0;
    sort(x.rbegin(), x.rend());
    ll r = 0;
    for (int i = 0; i < x.size(); i++) {
        if (r >= x[i].second) {
            r -= x[i].second;
            continue;
        }
        x[i].second -= r;
        ll tmp = (x[i].second + k - 1) / k;
        res += x[i].first * 2 * tmp;
        r = k * tmp - x[i].second;
    }
    return res;
}
```

Code

```
int main() {
    cin >> n >> k;
    ll x, m;
    for (int i = 0; i < n; i++) {
        cin >> x >> m;
        if (x > 0) a.push_back(pll(x, m));
        else b.push_back(pll(-x, m));
    }
    ll res = cal(a) + cal(b);
    cout << res << endl;
    return 0;
}
```

Thuật toán ứng dụng
Bài thực hành số 5.1: Tham lam

TS. Bùi Quốc Trung, TA. Đặng Xuân Vương

Trường Đại học Bách khoa Hà Nội
Viện Công nghệ thông tin và Truyền thông

Ngày 31 tháng 5 năm 2021