
# BÁNH MÌ SÀI GÒN

**Solution:**

| | |
|---|---|
| C++ | https://ideone.com/Er8Oz3 |
| Java | https://ideone.com/qy2mrE |
| Python | https://ideone.com/qBtsQi |

**Tóm tắt đề:**

Tèo cần mua bánh mì cho các bà con ở quê, thế nên anh ta ghi danh sách cần mua vào một quyển sổ nhưng khổ nỗi là lúc thì ghi số, lúc thì ghi chữ, rối tung cả lên. Ở tiệm bán bánh mì có một khuyến mãi cho những ai mua số lượng nhiều (số lượng 20 ổ trở lên), cứ mỗi lần mua 7 ổ thì giá mua mỗi ổ lần sau sẽ giảm thêm 10% giá tiền, số tiền giảm không vượt quá 50%.

Cho bạn danh sách người cần mua bánh mì, bạn hãy giúp Tèo tính tổng số lượng bánh mì cần mua và tổng tiền Tèo sẽ phải trả.

**Input**

Dòng đầu tiên ghi một số $N$ $(1 \leq N \leq 100.000)$ – số bà con gửi Tèo mua bánh mì.

N dòng tiếp theo mỗi dòng ghi: tên người mua (tên không dài quá 100 ký tự), giá trị cuối cùng của một dòng dữ liệu là số lượng bánh mì cần mua của người đó. Mỗi người không được mua quá 10 ổ bánh mì.

**Output**

In ra số nguyên K là tổng số bánh mì cần mua và một số thực (có một chữ số thập phân sau dấu phẩy) là giá tiền Tèo cần phải trả (cách nhau bởi dấu khoảng trắng).

**Ví dụ**

| | |
|---|---|
| 5<br>Di Nam 5<br>Co 7 Sau<br>Ong Noi 3<br>Co 8 8<br>Bac Hai Nam | 27 23.1 |

**Giải thích:**

Tổng số bánh mì mua 27 ổ:

- 7 ổ đầu mua giá 1 đồng: 7 đồng.
- 7 ổ sau mua giá 0.9 đồng: 6.3 đồng.
- 7 ổ sau mua giá 0.8 đồng: 5.6 đồng.
- 6 ổ sau mua giá 0.7 đồng: 4.2 đồng.

Tổng số tiền phải trả: 23.1 đồng

**Hướng dẫn giải:**

- Bước 1: Đọc từng dòng để xác định số lượng bánh mì mà mỗi người cần mua. Ta nhận thấy rằng ứng với mỗi dòng thì số lượng bánh mì sẽ nằm ở cuối, như vậy ta chỉ cần đọc ngược từ sau lên gặp khoảng trắng đầu tiên thì chuỗi ta vừa nhận được chính là số lượng bánh mì. Vì số lượng này có thể được biểu diễn bằng con số hay bằng chữ nên ta cần chú ý một chút: nếu như là con số thì sẽ đơn giản vì chỉ có thể là các trường hợp như “1”, “2”, “3”, …, “10” tuy nhiên nếu là chữ sẽ phức tạp hơn do có thể lẫn lộn các chữ in hoa, in thường trong đó, ví dụ như với số lượng là 1 có thể biểu diễn bằng chữ với nhiều cách như “mot”, “Mot”, “MOT”, “MoT”, “mOT”, … Thế nên ta cần chuẩn hóa chuỗi này về toàn chữ in hoa hay chữ in thường hết, sau đó mới ánh xạ qua các số nguyên. Đồng thời có thể tìm được tổng số lượng bánh mì cần mua trong bước này.
- Bước 2: Trước tiên cần xác định rõ tổng số lượng bánh mì có vượt quá 20 không:
    - Số lượng bánh mì ≤ 20: như vậy sẽ không áp dụng khuyến mãi nên tất cả các ổ bánh mì sẽ đều có giá là 1 đồng.
    - Số lượng bánh mì > 20: lúc này ta sẽ áp dụng khuyến mãi cứ mỗi lần mua 7 ổ thì giá mua lần sau sẽ giảm thêm 10% giá tiền, tuy nhiên số tiền giảm không vượt quá 50%. Ta có thể rút ra nhận xét như sau: ban đầu giá tiền mỗi ổ bánh mì là 1 đồng,

sau lần giảm thứ nhất là 0.9 đồng, lần thứ hai là 0.8 đồng, … cho đến khi giảm đến 0.5 (đã giảm đủ 50%) thì sẽ không giảm nữa. Như vậy ta chỉ cần lặp một vòng while thực hiện tăng tổng số tiền thêm 7 * (giá tiền bánh mì hiện tại), sau đó trừ số lượng bánh mì cần tính tiền đi 7, trừ giá tiền đi 0.1 cho đến khi số lượng bánh mì nhỏ hơn 7 hoặc giá tiền đã giảm chạm mốc 0.5. Bước cuối cùng là cộng thêm vào tổng tiền (số bánh mì còn lại chưa tính) * 0.5.

**Độ phức tạp:** **O(N*max(|M|))** với N là số bà con gửi Tèo mua bánh mì, |M| là độ dài chuỗi biểu diễn số lượng bánh mì cần mua cho mỗi người.

# BIỂN HIỆU SÀI GÒN

**Solution:**

| | |
|---|---|
| C++ | https://ideone.com/cbl0Pj |
| Java | https://ideone.com/Se1Xp1 |
| Python | https://ideone.com/6MKjHe |

**Tóm tắt đề**:

Bạn được cho một dãy số nguyên dương a có N phần tử và Q truy vấn. Với mỗi truy vấn bạn nhận được một số nguyên dương d, hãy đếm xem có bao nhiêu phần tử trong dãy a nhận d làm ước của chúng.

**Input**

- Dòng đầu tiên gồm một số nguyên dương N là số lượng phần tử. ($1 \le N \le 10^6$).
- Dòng thứ 2 gồm N số nguyên dương, số nguyên dương thứ i có giá trị là $a_i$ tương ứng là phần tử thứ i của dãy a ($1 \le a_i \le 10^6$).
- Dòng thứ 3 gồm 1 số nguyên dương Q là số lượng truy vấn. ($1 \le Q \le 10^5$).
- Q dòng tiếp theo, mỗi dòng gồm một số nguyên dương d tương ứng với câu hỏi: "Có bao nhiêu số trong dãy a nhận d làm ước của chúng". ($1 \le d \le \max\{a_1, a_2,...,a_N\}$) trong đó $\max\{a_1, a_2,...,a_N\}$ là giá trị lớn nhất của các phần tử trong N phần tử của dãy a.

**Output**

Gồm Q dòng, mỗi dòng sẽ tương ứng là một câu trả lời cho truy vấn tương ứng.

**Ví dụ**

| | |
|---|---|
| 5<br>2 4 1 5 10<br>5<br>1<br>3<br>2<br>10<br>2 | 5<br>0<br>3<br>1<br>3 |

**Giải thích:**

- Với truy vấn thứ 1, toàn bộ các số trong dãy đều chia hết cho 1.
- Với truy vấn thứ 2, không có số nào chia hết cho 3.

- Với truy vấn thứ 3, có các số {2, 4, 10} chia hết cho 2.
- Với truy vấn thứ 4, có số 10 chia hết cho 10.
- Với truy vấn thứ 5, ta có kết quả giống truy vấn thứ 3.

**Hướng dẫn giải:**

Sử dụng mảng b với b[x] là số lần xuất hiện của giá trị x trong mảng a, mảng cnt với cnt[x] là số bội số của x trong mảng a. Như vậy với mỗi truy vấn d ta chỉ cần in ra cnt[d].

Để xây dựng được mảng cnt, ta sẽ duyệt i từ 1 đến $10^6$ (giá trị tối đa có thể xuất hiện trong mảng), với mỗi i, cnt[i] sẽ được tính theo công thức cnt[i] = cnt[i] + b[j] với j là các bội số của i.

**Độ phức tạp:** Gọi MAX = $10^6$, thì ta thấy rằng số bội của i sẽ bằng $\frac{MAX}{i}$. Vậy để xây dựng được mảng cnt là $\frac{MAX}{1} + \frac{MAX}{2} + \cdots + \frac{MAX}{MAX} \simeq MAX * log_2(MAX)$. Với mỗi truy vấn mình chỉ cần trả lời trong O(1). Vậy tổng độ phức tạp là $\boldsymbol{O(MAX * log_2(MAX) + Q)}$

# SÀI GÒN NGÀY NAY

**Solution:**

| | |
|---|---|
| C++ | https://ideone.com/FlNyl3 |
| Java | https://ideone.com/2DsZah |
| Python | https://ideone.com/fuGAzU |

**Tóm tắt đề:**

Bạn là chủ một cửa hàng xe đạp. Mỗi tháng bạn đều bán hết xe và sau đó chọn nhập hàng từ một số đại lý trong N đại lý. Nếu bạn chọn nhập xe từ đại lý thứ i thì bạn phải nhập đúng bi chiếc và với mỗi chiếc bạn có thể bán với lãi tối đa là ai đồng. Bạn hãy chọn nhập xe từ nhiều đại lý nhất sao cho lãi suất trung bình trên mỗi chiếc xe bạn bán ra là ít nhất A đồng.

**Input:**

Dòng đầu tiên chứa 2 số nguyên dương N và A được ngăn cách nhau bởi một khoảng trắng là số lượng đại lý bạn có thể chọn để nhập hàng ($N \leq 10^5$) và lãi suất trung bình tối thiểu bạn cần đạt được ($0 \leq A \leq 10^9$).

Dòng thứ 2 gồm N số nguyên dương $a_1, a_2, \ldots, a_N$ ($a_i \leq 10^6$) là lãi suất tối đa mà bạn có thể thu được khi bán mỗi chiếc xe đạp của từng đại lý.

Dòng thứ 3 gồm N số nguyên dương $b_1, b_2, \ldots, b_N$ ($b_i \leq 10^6$) là số xe mà bạn sẽ mua ở mỗi đại lý.

**Output:**

In ra 'FAILED' nếu tháng này bạn không thể đạt lãi suất trung bình trên mỗi chiếc xe là A đồng, ngược lại in ra số hãng xe nhiều nhất mà bạn có thể chọn để nhập xe về bán.

**Ví dụ:**

| | |
|---|---|
| 3 5<br>1 2 6<br>1 4 10 | 2 |

| | |
|---|---|
| 1 100<br>30 | FAILED |

**Giải thích:**

**Ví dụ 1**: Bạn có 3 đại lý xe để chọn.

- Nếu bạn chọn mua xe từ đại lý thứ 1 và thứ 3 thì doanh thu của bạn sẽ là $1 \times 1 + 6 \times 10 = 61$ đồng và số xe bạn bán được là $1 + 10 = 11$, lãi suất trung bình mỗi chiếc xe bán ra là $\frac{61}{11} \approx 5.55$ đồng.
- Nếu bạn mua ở cả 3 đại lý luôn thì lãi suất trung bình đạt được là $\frac{1\times1+2\times4+6\times10}{1+4+10} = 4.6$ đồng, không đạt mục tiêu của bạn.

**Ví dụ 2**: Bạn chỉ có thể đặt xe từ 1 đại lý và lãi suất trung bình trên mỗi chiếc xe bán ra là 30 đồng, không đạt mục tiêu của bạn.

**Hướng dẫn giải:**

Giả sử bạn chọn các đại lý $i_1$, $i_2$, …, $i_k$.

Điều kiện để lãi suất trung bình trên mỗi chiếc xe bán ra đạt ít nhất A đồng là $\frac{\sum b[i_j]\times a[i_j]}{\sum b[i_j]} \geq A$.

Điều này tương đương với:

$$\sum b\left[i_j\right] \times a\left[i_j\right] \geq A \times \sum b\left[i_j\right]$$
$$\Leftrightarrow \sum b\left[i_j\right] \times \left(a\left[i_j\right] - A\right) \geq 0$$

Như vậy bạn chỉ cần chọn nhiều hãng xe nhất sao cho tổng các giá trị $b \times (a - A)$ của các hãng bạn chọn là không âm. Điều này có thể được thực hiện đơn giản bằng cách tham lam như sau: sắp xếp các hãng xe lại theo giá trị $b \times (a - A)$, sau đó lần lượt chọn các hãng có$b \times (a - A)$ lớn nhất cho đến khi tổng các giá trị này bé hơn 0 thì dừng.

**Độ phức tạp:** Thao tác tốn chi phí nhất là sắp xếp N hãng xe và nó có chi phí về thời gian là O(NlogN) (nếu bạn sử dụng các hàm sắp xếp có sẵn của các ngôn ngữ). Vậy độ phức tạp của thuật toán này là **O(NlogN)**.

# SÀI GÒN 2518

**Solution:**

| | |
|---|---|
| C++ | https://ideone.com/Jt6ibW |
| Java | https://ideone.com/W4eUJg |
| Python | https://ideone.com/EZtMas |

**Tóm tắt đề:**

Cho một đơn đồ thị vô hướng gồm $N$ đỉnh được đánh số từ 0 đến $N-1$ và $M$ cạnh, đỉnh $i$ có giá trị $a_i$. Thực hiện $Q$ truy vấn, truy vấn thứ $i$ yêu cầu: với tất cả các đỉnh $v$ có khoảng cách đến đỉnh $u_i$ nhỏ hơn $d_i$, gán $a_v = \max(a_v, x_i)$. Cho biết giá trị của các đỉnh sau khi thực hiện $Q$ truy vấn trên.

**Input:**

Dòng đầu tiên ghi ba số $N, M, Q$ $(1 \le N, M, Q \le 200000)$ – số đỉnh, số cạnh và số truy vấn.

$M$ dòng tiếp theo, mỗi dòng gồm hai số nguyên $u$ và $v$ $(0 \le u, v < N)$ mô tả một cạnh trong đồ thị.

Q dòng tiếp theo, dòng thứ $i$ gồm ba số nguyên $u_i, d_i, x_i$ $(0 \le u_i < N, 0 \le d_i \le 20, 1 \le x_i \le 10^9)$ mô tả truy vấn thứ $i$.

**Output:**

In ra $N$ số nguyên là giá trị của các đỉnh sau khi thực hiện xong $Q$ truy vấn.

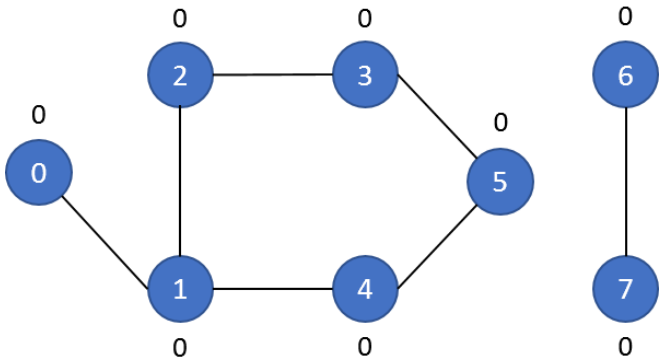
**Ví dụ:**

| | |
|---|---|
| 8 7 4<br>0 1<br>1 2<br>1 4<br>2 3<br>3 5<br>4 5<br>6 7<br>1 0 4<br>0 2 2<br>3 0 5<br>6 5 1 | 2 4 2 5 2 0 1 1 |

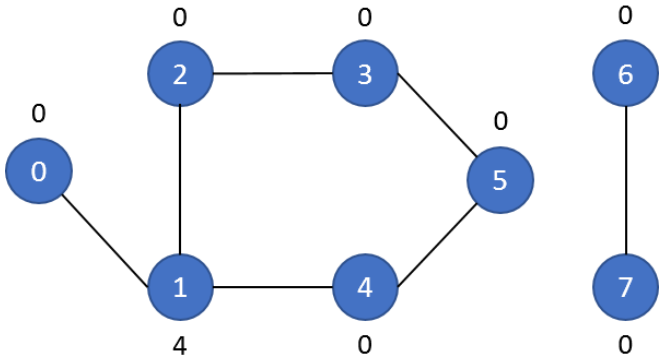
| | |
|---|---|
| 1 100<br>30 | FAILED |

**Giải thích:**

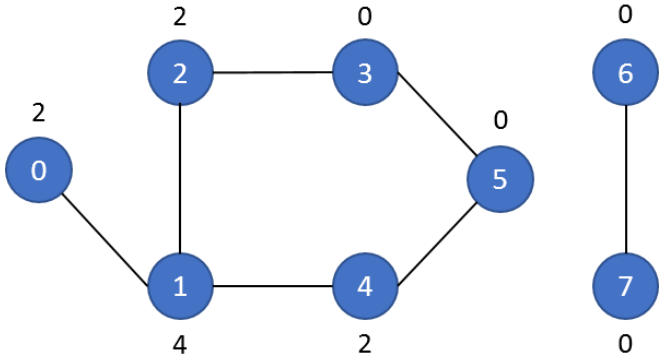
- Hình vẽ mô tả đất nước Berland ban đầu



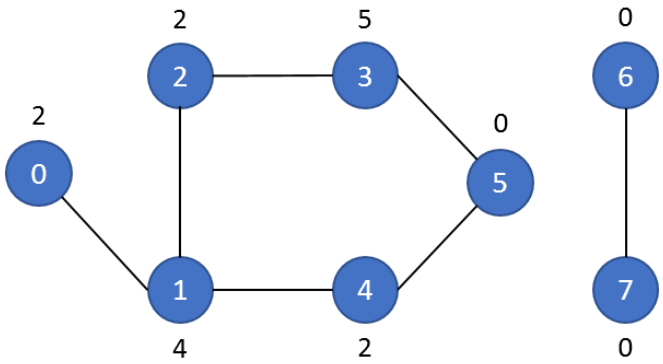

- Sau chính sách thứ nhất



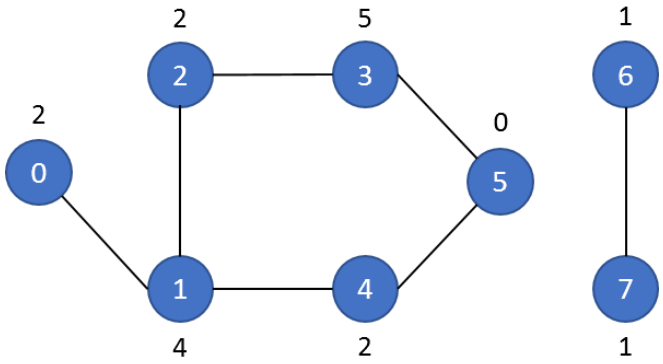

- Sau chính sách thứ hai




- Sau chính sách thứ ba




- Sau chính sách thứ tư




**Hướng dẫn giải:**

Ta sẽ thực hiện các truy vấn theo thứ tự $x_i$ giảm dần. Khi đó, yêu cầu truy vấn trở thành: với tất cả các đỉnh $v$ có khoảng cách đến đỉnh $u_i$ nhỏ hơn $d_i$, nếu $a_v = 0$ thì gán $a_v = x_i$.

Khi đó, mỗi truy vấn $(u_i, d_i, c_i)$ có thể được bằng thuật toán tương tự DFS như sau:

- Khởi tạo $a_u = 0$ ($a_u$ là giá trị của đỉnh $u$).
- Dùng hàm đệ quy với ba tham số $u$, $d$ và $x$ (với $u$ là đỉnh hiện tại, $d$ là khoảng cách còn lại có thể đi được, $x$ là giá trị cần gán).
    - Nếu đã đi đến trạng thái $(u, d)$ thì tức là các đỉnh có khoảng cách $d$ đến $u$ đều đã được gán giá trị nên ta có thể thoát khỏi hàm ngay.
    - Nếu $d < 0$, nghĩa là khoảng cách từ đỉnh hiện tại đến đỉnh xuất phát đã vượt quá giới hạn của truy vấn, ta thoát khỏi hàm.
    - Ngược lại:
        - Đánh dấu là đã đi đến trạng thái $(u, d)$
        - Thực hiện gán giá trị cho $a_u$ (nếu $a_u = 0$)
        - Duyệt tất cả các đỉnh kề với $u$ và gọi hàm đệ quy với ba tham số $(v, d - 1, x)$.

- Để thực hiện truy vấn thứ $i$, ta gọi hàm đệ quy với ba tham số $(u_i, d_i, x_i)$.

```
FUNCTION dfs(u, d, x)
BEGIN
    IF đã đi đến trạng thái (u, d) OR d < 0
        RETURN
    Đánh dấu đã đi đến trạng thái (u, d)
    IF a[u] = 0
        a[u] = x
    FOR v kề với u
        dfs(v, d-1, x)
END

FOR EACH truy vấn i
    dfs(u[i], d[i], x[i])
```

**Độ phức tạp:** **O(Q log Q + (N+M)*D)** với $N$ là số đỉnh, $M$ là số cạnh, $Q$ là số lượng truy vấn, $D$ là khoảng cách $d_i$ lớn nhất trong các truy vấn.