

# Nộp lời giải assignment 5

Nguyễn Trường Giang

MSSV: 24520011

## Bài 1: Alpha's Problem

Đề bài:

Cho một số nguyên  $a$  trong hệ thập phân và yêu cầu chuyển đổi số đó sang hệ cơ số  $x$  ( $2 \leq x \leq 36$ ).

Kết quả là biểu diễn của  $a$  trong hệ cơ số  $x$ .

**Input:**

- Một dòng duy nhất chứa hai số nguyên  $a$  ( $0 \leq a \leq 10^{18}$ ) và  $x$  cách nhau bởi khoảng trắng.

**Output:**

- Một dòng duy nhất là biểu diễn của  $a$  trong hệ cơ số  $x$ .

**Ví dụ:**

Input	Output
12345 6	133053
987654321 2	111010110111100110100010110001

**Lời giải:**

Để chuyển đổi số  $a$  từ hệ thập phân sang hệ cơ số  $x$ , ta thực hiện như sau:

- Lấy phần dư của  $a$  khi chia cho  $x$  để xác định chữ số cuối cùng trong hệ cơ số  $x$ .

2. Chia `a` cho `x` để giảm giá trị của `a`.
  3. Lặp lại quá trình trên cho đến khi `a` bằng 0.
  4. Đảo ngược thứ tự các chữ số thu được để có kết quả đúng.
- 

Code full:

```
#include<bits/stdc++.h>
#define fi first
#define se second
using namespace std;
using ll = long long;
using ii = pair<int, int>

const int MOD = 1e9 + 7;
const int MAX = 1e5 + 5;

#define TASK ""

int main() {
    ios::sync_with_stdio(0); cin.tie(0); cout.tie(0);

    int num; cin >> num;
    int x; cin >> x;

    vector<int> ans;

    while (num > 0) {
        ans.push_back(num % x);
        num /= x;
    }
    reverse(ans.begin(), ans.end());
    for (int x : ans) cout << x;

    return 0;
}
```

## Bài 2: Nối chữ

Đề bài:

Cho một chuỗi `s` chỉ gồm các chữ cái tiếng Anh viết thường. Bạn được phép thực hiện thao tác sau đây không giới hạn số lần:

- Chọn hai ký tự giống nhau liên tiếp trong chuỗi và xóa chúng.

Yêu cầu: Tính độ dài ngắn nhất của chuỗi  $s$  sau khi thực hiện các thao tác trên.

**Input:**

- Một dòng duy nhất chứa chuỗi  $s$  ( $1 \leq |s| \leq 2 \times 10^5$ ).

**Output:**

- Một số nguyên không âm duy nhất là độ dài ngắn nhất của chuỗi sau khi thực hiện các thao tác.

**Ví dụ:**

Input	Output
aaabccddd	3
baab	0

**Lời giải:**

Để giải bài toán, ta sử dụng cấu trúc dữ liệu **stack**:

- Duyệt từng ký tự trong chuỗi  $s$ .
- Nếu stack không rỗng và ký tự trên đỉnh stack giống ký tự hiện tại, ta xóa ký tự trên đỉnh stack (tức là xóa cặp ký tự giống nhau).
- Nếu không, ta đẩy ký tự hiện tại vào stack.
- Sau khi duyệt hết chuỗi, độ dài của stack chính là độ dài ngắn nhất của chuỗi sau khi thực hiện các thao tác.

**Code full:**

```
#include<bits/stdc++.h>
#define fi first
#define se second
using namespace std;
using ll = long long;
using ii = pair<int, int>;
```

```

const int MOD = 1e9 + 7;
const int MAX = 2e5 + 5;

#define TASK ""

ll n, s;
int a[MAX];

int main() {
    ios::sync_with_stdio(0); cin.tie(0); cout.tie(0);
    cin >> n >> s;
    for (int i = 1; i <= n; i++) {
        cin >> a[i];
    }

    priority_queue<ll, vector<ll>, greater<ll>> pq;

    for (int i = 1; i <= n; i++) {
        pq.push(a[i]);
    }

    ll ans(0);
    while (pq.size() != 1) {
        ll a = pq.top(); pq.pop();
        ll b = pq.top(); pq.pop();

        ans += a + b;
        pq.push(a + b);
    }

    cout << ans << endl;

    return 0;
}

```

## Bài 3: Cắt gỗ

Đề bài:

Ông Hai có một khúc gỗ có chiều dài S. Ông muốn cắt khúc gỗ thành n phần với chiều dài lần lượt là  $a_1, a_2, \dots, a_n$ .

Chi phí để chia khúc gỗ thành hai phần là chiều dài của khúc gỗ đó.

Ví dụ: Ông Hai muốn chia khúc gỗ có chiều dài 16 thành ba phần 1, 9, 6.

- Bước một: cắt khúc gỗ có kích thước 16 thành hai phần 9 và 7 với chi phí là 16.
- Bước hai: cắt khúc gỗ kích thước 7 thành hai phần 6 và 1 với chi phí là 7.

Tổng chi phí cho cách cắt này là  $16 + 7 = 23$ , đồng thời đây là cách cắt tối ưu cho chi phí thấp nhất.

**Yêu cầu:** Hãy giúp Ông Hai tính chi phí cắt gỗ thấp nhất.

**Input:**

- Dòng đầu gồm hai số nguyên dương  $n$  và  $s$  ( $1 \leq n \leq 2 \times 10^5$ ).
- Dòng sau gồm  $n$  số nguyên dương  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^9$ ).  
Input đảm bảo  $\sum a_i = s$ .

**Output:**

- Gồm một số nguyên dương duy nhất là chi phí cắt gỗ thấp nhất.

**Ví dụ:**

Input	Output
3 16	23
1 6	0
5 24	52
8 1 3 5 7	17

**Lời giải:**

Để giải bài toán, ta sử dụng cấu trúc dữ liệu **priority queue** (hàng đợi ưu tiên) để tối ưu hóa chi phí cắt gỗ:

- Đưa tất cả các đoạn gỗ vào priority queue (min-heap).
- Lặp lại cho đến khi chỉ còn một đoạn gỗ trong hàng đợi:
- Lấy hai đoạn gỗ có chiều dài nhỏ nhất ra khỏi hàng đợi.
- Tính chi phí cắt gỗ bằng tổng chiều dài của hai đoạn gỗ này.
- Đưa đoạn gỗ mới (sau khi gộp) vào lại hàng đợi.
- Tổng chi phí cắt gỗ là tổng các chi phí tính được ở bước trên.

---

## Code full:

```

#include<bits/stdc++.h>
#define fi first
#define se second
using namespace std;
using ll = long long;
using ii = pair<int, int>

const int MOD = 1e9 + 7;
const int MAX = 5e5 + 5;

#define TASK ""

int n;
int num[MAX];
int len[MAX];

void Init() {
    cin >> n;
    for (int i = 1; i <= n; i++) {
        cin >> num[i];
    }

    map<int, int> cnt;
    for (int i = 1; i <= n; i++) {
        cnt[num[i]]++;
    }

    n = 0;
    for (auto it : cnt) {
        num[++n] = it.se;
        len[n] = it.fi;
    }
}

int main() {
    ios::sync_with_stdio(0); cin.tie(0); cout.tie(0);
    Init();

    ll ans(0);

    int j = 1, sum = 0;
    for (int i = 1; i <= n; i++) {
        while (j <= n && len[j] < 2LL * len[i]) {
            sum += num[j];
            j++;
        }

        ans += 1LL * num[i] * (num[i] - 1) * (num[i] - 2) / 6;
    }
}

```

```

        ans += 1LL * num[i] * (num[i] - 1) / 2 * (sum - num[i]);
    }

    cout << ans << endl;

    return 0;
}

```

## Bài 4: Tam giác cân

### Đề bài:

Tam giác cân là tam giác có ít nhất 2 cạnh có độ dài bằng nhau.

Cho dãy gồm  $N$  số nguyên dương:  $a_1, a_2, \dots, a_n$ . Hãy tính số bộ 3 chỉ số  $(i, j, k)$ , với  $1 \leq i < j < k \leq N$  sao cho 3 số  $a_i, a_j, a_k$  là độ dài 3 cạnh của một tam giác cân.

### Input:

- Dòng đầu ghi số nguyên  $N$  ( $3 \leq N \leq 5 \times 10^5$ ).
- Dòng tiếp theo ghi  $N$  số nguyên dương là các phần tử của dãy, giá trị không vượt quá  $10^9$ .

### Output:

- Gồm một dòng chứa số lượng bộ ba chỉ số phân biệt  $(i, j, k)$ , với  $1 \leq i < j < k \leq N$  sao cho 3 số  $a_i, a_j, a_k$  là độ dài 3 cạnh của một tam giác cân.

### Ví dụ:

Input	Output
8	22
5 3 2 9 5 4 9 5	

### Lời giải:

Để giải bài toán, ta cần kiểm tra điều kiện để 3 cạnh tạo thành tam giác cân:

1. Điều kiện để 3 số  $a_i, a_j, a_k$  tạo thành tam giác:

$$2. a_i + a_j > a_k$$

$$3. a_i + a_k > a_j$$

$$4. a_j + a_k > a_i$$

5. Điều kiện để tam giác cân:

6. Có ít nhất 2 cạnh bằng nhau.

### Ý tưởng:

1. Sắp xếp mảng  $a$  theo thứ tự tăng dần.
2. Sử dụng 2 vòng lặp để chọn 2 cạnh độ dài  $a[i]$  và một cạnh độ dài  $a[j]$ .
3. Để tối ưu từ  $O(n^2)$  xuống còn  $O(n \log n)$  thì tại vòng lặp thứ 2 để tìm  $j$  thỏa mãn, ta sử dụng tìm kiếm nhị phân để giảm độ phức tạp.
4. Lưu ý: Cần xét thêm trường hợp tam giác đều.

### Độ phức tạp:

- Sắp xếp mảng:  $O(N \log N)$ .

### Code full:

```
#include<bits/stdc++.h>
#define fi first
#define se second
using namespace std;
using ll = long long;
using ii = pair<int, int>

const int MOD = 1e9 + 7;
const int MAX = 5e5 + 5;

#define TASK ""

int n;
int num[MAX];
int len[MAX];

void Init() {
    cin >> n;
    for (int i = 1; i <= n; i++) {
        cin >> num[i];
    }
}
```

```

map<int, int> cnt;
for (int i = 1; i <= n; i++) {
    cnt[num[i]]++;
}

n = 0;
for (auto it : cnt) {
    num[++n] = it.se;
    len[n] = it.fi;
}
}

int main() {
    ios::sync_with_stdio(0); cin.tie(0); cout.tie(0);

    Init();

    ll ans(0);

    int j = 1, sum = 0;
    for (int i = 1; i <= n; i++) {
        while (j <= n && len[j] < 2LL * len[i]) {
            sum += num[j];
            j++;
        }

        ans += 1LL * num[i] * (num[i] - 1) * (num[i] - 2) / 6;
        ans += 1LL * num[i] * (num[i] - 1) / 2 * (sum - num[i]);
    }

    cout << ans << endl;

    return 0;
}

```

## Bài 5: Chênh lệch

### Đề bài:

Thành phố Alpha có  $n$  tòa nhà cao tầng, được đánh số từ 1 đến  $n$ . Tòa nhà cao tầng thứ  $i$  có độ cao là  $a_i$ .

Độ chênh lệch độ cao giữa hai tòa nhà  $i, j$  là hiệu số độ cao giữa hai tòa nhà đó, hay nói cách khác là  $|a_i - a_j|$ .

Để phục vụ cho công tác quản lý thành phố, chính quyền cần tính tổng độ chênh lệch giữa từng đôi một tòa nhà. Hay nói cách khác là:

$$\sum |a_i - a_j| \text{ với } 1 \leq i < j \leq n$$

Hãy tính kết quả mà chính quyền cần.

**Input:**

- Dòng đầu tiên chứa số nguyên  $n$  là số tòa nhà cao tầng trong thành phố ( $1 \leq n \leq 2 \times 10^5$ ).
- Dòng thứ hai chứa  $n$  số nguyên bao gồm các số  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^9$ ).

**Output:**

- Gồm một số duy nhất là kết quả cần tìm.

**Ví dụ:**

Input	Output
3	194
100 3 7	

**Lời giải:**

Ta sử dụng ý tưởng chính: Loại bỏ dấu giá trị tuyệt đối bằng cách biết sắp xếp mảng, từ đó biết trước giá trị nào lớn hơn giá trị nào.

1. Sắp xếp mảng  $a$  theo thứ tự tăng dần.
2. Với mỗi tòa nhà  $i$ , tính tổng độ chênh lệch của nó với các tòa nhà đứng trước và sau:
3. Độ chênh lệch với các tòa nhà trước:  $(i - 1) * a[i] - \text{tổng độ cao của các tòa nhà trước}$ .
4. Độ chênh lệch với các tòa nhà sau:  $\text{tổng độ cao của các tòa nhà sau} - (n - i) * a[i]$ .
5. Cộng dồn các giá trị trên để tính tổng độ chênh lệch.

**Độ phức tạp:**

- Sắp xếp mảng:  $O(n \log n)$ .
- Tính toán tổng độ chênh lệch:  $O(n)$ .

---

Code full:

```
#include<bits/stdc++.h>
#define fi first
#define se second
using namespace std;
using ll = long long;
using ii = pair<int, int>

const int MOD = 1e9 + 7;
const int MAX = 2e5 + 5;

int n;
int a[MAX];

int main() {
    ios::sync_with_stdio(0); cin.tie(0); cout.tie(0);

    cin >> n;
    for (int i = 1; i <= n; i++) {
        cin >> a[i];
    }

    sort(a + 1, a + n + 1);
    ll ans = 0, sum = 0;
    for (int i = 1; i <= n; i++) {
        ans += (ll)(i - 1) * a[i] - sum;
        sum += a[i];
    }

    cout << ans << endl;

    return 0;
}
```

Bài 6:

Bài 6: Tom và tiền tiết kiệm

Đề bài:

Sau một thời gian dài tích góp, Tom đang có trong tay  $n$  tờ tiền, tờ tiền thứ  $i$  mang mệnh giá  $v_i$ .

Một ngày đẹp trời, Tom quyết định đem hết số tiền bản thân tích góp được đi gửi ngân hàng với ước mơ trở thành tỷ phú.

Tình cờ thay, hôm đó là ngày đầu tiên Jerry đến nhận việc tại ngân hàng và Tom chính là vị khách hàng đầu tiên mà Jerry phục vụ. Để tránh việc nhầm lẫn không đáng có, việc đầu tiên Jerry cần làm là tính số lượng mệnh giá tiền phân biệt mà Tom gửi vào ngân hàng.

Do Jerry là nhân viên mới, còn thiếu kinh nghiệm nên mọi người hãy giúp Jerry nhé!

### Input:

- Dòng đầu tiên chứa số  $n$  là số lượng tờ tiền mà Tom có. ( $1 \leq n \leq 3 \times 10^5$ )
- Dòng thứ hai chứa  $n$  số nguyên dương, lần lượt là  $v_1, v_2, \dots, v_n$ . ( $1 \leq v_i \leq 10^9$ )

### Output:

- Dòng đầu ghi số nguyên dương  $k$  là số mệnh giá tiền phân biệt mà Tom gửi vào.
- Dòng thứ hai ghi  $k$  số nguyên dương  $b_1, b_2, \dots, b_k$ , là các mệnh giá tiền được sắp xếp theo thứ tự tăng dần.

### Ví dụ:

Input	Output
6	4
3 100 2 3 3 10	2 3 10 100

### Lời giải:

Để giải bài toán, ta cần tìm các mệnh giá tiền phân biệt và sắp xếp chúng theo thứ tự tăng dần:

1. Sử dụng cấu trúc dữ liệu **set** để lưu các mệnh giá tiền. Set tự động loại bỏ các phần tử trùng lặp và sắp xếp các phần tử theo thứ tự tăng dần.
2. Duyệt qua từng mệnh giá tiền trong danh sách  $v$  và thêm vào set.
3. In ra kích thước của set (số lượng mệnh giá phân biệt) và các phần tử trong set.

### Độ phức tạp:

- Thêm  $n$  phần tử vào set:  $O(n \log n)$ .
- In ra các phần tử trong set:  $O(k)$ , với  $k$  là số mệnh giá phân biệt.

Code full:

```
#include<bits/stdc++.h>
#define fi first
#define se second
using namespace std;
using ll = long long;
using ii = pair<int, int>

const int MOD = 1e9 + 7;
const int MAX = 3e5 + 5;

int main() {
    ios::sync_with_stdio(0); cin.tie(0); cout.tie(0);

    int n; cin >> n;

    set<int> s;
    while (n--) {
        int x; cin >> x;
        s.insert(x);
    }

    cout << s.size() << endl;
    for (int x : s) {
        cout << x << " ";
    }

    return 0;
}
```

## Bài 7: Đèn hoa

Đề bài:

Để chuẩn bị một ngày lễ lớn lần thứ  $k$ , thành phố quyết định dùng đèn LED kết thành  $k$  bông hoa trang trí dọc đường phố chính, mỗi bông hoa được kết từ một loại bóng LED cùng màu.

Để có hiệu ứng ánh sáng tốt nhất, Công ty Chiếu sáng được yêu cầu tạo ra được càng nhiều hoa khác màu càng tốt. Trong kho của Công ty có  $n$  bộ đèn LED, mỗi bộ lắp được một đèn và bộ thứ  $i$  có màu  $a_i$ .

Hãy chỉ ra  $k$  màu của các bộ đèn cần chọn sao cho số lượng hoa khác màu lắp được là nhiều nhất. Nếu có nhiều cách khác nhau thì đưa ra cách tùy chọn.

#### Input:

- Dòng đầu tiên chứa 2 số nguyên  $n$  và  $k$  ( $1 \leq k \leq n \leq 10^5$ ).
- Dòng thứ hai chứa  $n$  số nguyên  $a_1, a_2, \dots, a_n$  ( $1 \leq a_i \leq 10^9, i = 1 \div n$ ).

#### Output:

- Gồm một dòng chứa  $k$  số nguyên – màu của các đèn được chọn.

#### Ví dụ:

#### Input:

```
10 4
8 8 8 8 8 8 8 2 1
```

#### Output:

```
1 2 8 8
```

#### Lời giải:

Để giải bài toán, ta cần chọn  $k$  màu sao cho số lượng hoa khác màu là nhiều nhất. Nếu không đủ  $k$  màu khác nhau, ta có thể chọn thêm các màu đã chọn trước đó.

#### Ý tưởng:

1. Sử dụng cấu trúc dữ liệu **map** để đếm số lượng mỗi màu trong danh sách  $a$ .
2. Sắp xếp các màu theo số lượng giảm dần.
3. Chọn tối đa  $k$  màu từ danh sách đã sắp xếp:
4. Nếu số lượng màu khác nhau đủ  $k$ , chọn  $k$  màu đầu tiên.
5. Nếu không đủ, chọn tất cả các màu khác nhau, sau đó bổ sung thêm các màu đã chọn trước đó để đủ  $k$ .

#### Độ phức tạp:

- Đếm số lượng mỗi màu:  $O(n)$ .

- Sắp xếp các màu:  $O(m \log m)$ , với  $m$  là số lượng màu khác nhau.
- Chọn  $k$  màu:  $O(k)$ .

Code full:

```
vector<int> get_ans(const vector<int>& A, int K){
    map<int,int> Hash;
    vector<int> ans;
    for (int i = 0; i < A.size(); i++){
        Hash[A[i]]++;
    }

    for (int rep = 1; rep <= n; rep++){
        vector<int> list_erase;

        for (pair<const int,int> &p: Hash){
            p.second--;
            if (p.second == 0){
                list_erase.push_back(p.first);
            }

            if (K > 0) {
                ans.push_back(p.first);
                K--;
            }
            else {
                break;
            }
        }
        for (int i = 0; i < list_erase.size(); i++){
            Hash.erase(list_erase[i]);
        }
    }
    return ans;
}
```

## Bài 8: Kiểm kê

Đề bài:

Cửa hàng X có nhiều loại hàng, mỗi loại hàng đều có mã khác nhau. Tuy nhiên, trong lúc bán hàng, nhân viên quên nhập số lượng mỗi loại mà chỉ nhập mã các loại hàng mà các loại hàng trên từng dòng.

Bạn hãy đếm xem có bao nhiêu loại hàng khác nhau?

**Ví dụ:**

Có một số loại hàng có mã lần lượt là [123, 45, 8, 19, 1, 2333].

Bạn nhận viên sẽ nhập theo từng dòng 123, 45, 45, 8, 1, 123, 123, 8, 8.

Số loại hàng khác nhau trong đây được nhập sẽ là 4.

**Input:**

- Dòng đầu tiên là số nguyên  $N$  ( $1 \leq N \leq 5 \times 10^4$ ) là số lượng mã loại hàng được nhập.
- $N$  dòng tiếp theo, mỗi dòng là một mã của loại hàng được nhập, độ dài của mã có thể lên tới 100.

**Output:**

- Một dòng chứa số nguyên là số các loại hàng khác nhau.

**Ví dụ:**

Input	Output
10	7
123	
123444	
1234444	
45	
1	
2333	
45	
455	
455	

Input	Output
1	

### Lời giải:

Để giải bài toán, ta cần đếm số lượng mã loại hàng khác nhau.

1. Sử dụng cấu trúc dữ liệu **set** để lưu các mã loại hàng. Set tự động loại bỏ các phần tử trùng lặp.
2. Duyệt qua từng mã loại hàng trong danh sách nhập và thêm vào set.
3. Kích thước của set chính là số lượng mã loại hàng khác nhau.

### Độ phức tạp:

- Thêm  $N$  phần tử vào set:  $O(N \log N)$ .

### Code full:

```
int count_distinct(const vector<string>& ids){  
  
    map<string, int> Hash;  
  
    for (int i = 0; i < N; i++) {  
        Hash[ids[i]]++;  
    }  
    return Hash.size();  
}
```