

## TỔNG QUAN ĐỀ THI

Bài	Tên bài	Tên file	Tên file input	Tên file output	Điểm
1	Tìm ước chung lớn nhất	UCLN.*	UCLN.INP	UCLN.OUT	5
2	Tìm số nguyên tố lớn nhất	NTMAX.*	NTMAX.INP	NTMAX.OUT	7
3	Dãy con liên tiếp có tổng chia hết cho k	CHAK.*	CHIAK.INP	CHIAK.OUT	5
4	Tặng quà	QUA.*	QUA.INP	QUA.OUT	3

Dấu \* là PY hoặc CPP tùy theo ngôn ngữ lập trình là PYTHON hay C++

### BÀI 1. TÌM ƯỚC CHUNG LỚN NHẤT

**\* Thuật toán đề xuất:**

Một bài toán rất quen thuộc, bạn đọc tự giải.

Lưu ý: vì  $N, M \leq 10^{12}$  nên cần khai báo N, M kiểu long long đối với ngôn ngữ lập trình C++

**Chủ đề: Duyệt**

**\* Chương trình minh họa bằng ngôn ngữ lập trình C++:**

```
#include <bits/stdc++.h>

using namespace std;
long long N, M;
int main()
{
    freopen("UCLN.INP", "r", stdin);
    freopen("UCLN.OUT", "w", stdout);
    cin >> N >> M;
    cout << __gcd(N, M);
    return 0;
}
```

**\* Chương trình minh họa bằng ngôn ngữ lập trình Python:**

```
import sys, math
sys.stdin = open("ucln.inp", "r")
sys.stdout = open("ucln.out", "w")
n, m = map(int, input().split())
print(math.gcd(n, m))
```

## BÀI 2. TÌM SỐ NGUYÊN TỐ LỚN NHẤT

### \* Thuật toán đề xuất:

Duyệt qua từng ký tự:

- nếu gặp ký tự số thì tăng biến đếm số lượng chữ số, đồng thời bổ sung vào số  $p$
- nếu gặp ký tự khác số thì kiểm tra  $p$  nguyên tố và cập nhật lại số nguyên tố max, đồng thời gán lại giá trị  $p = 0$ .

Lưu ý: để kiểm tra số nguyên tố nhanh cần sử dụng thuật toán sàng số nguyên tố.

Độ phức tạp:

- Độ phức tạp sàng số nguyên tố:  $O(5 * 10^6 * \log(5 * 10^6))$
- Độ phức tạp xử lý tìm  $p$ :  $O(n)$
- Độ phức tạp cuối cùng:  $O(5 * 10^6 * \log(5 * 10^6))$

Chủ đề: duyệt + số học

### \* Chương trình minh họa bằng ngôn ngữ lập trình C++:

```
#include <bits/stdc++.h>
#define N int(5e6)
using namespace std;
string s;
int c[N+3];
int pmax = 0, p;
void sangngto()
{
    fill(c+2,c+N+1,1);
    for (int i=2; i*i <= N; i++)
        if (c[i] == 1)
            for (int j=i*i; j<=N; j+=i) c[j] = 0;
}
int main()
{
    ios_base::sync_with_stdio(0);
    cin.tie(0); cout.tie(0);
    freopen("ntmax.inp","r",stdin);
    freopen("ntmax.out","w",stdout);
    sangngto();
    cin >> s;
    int dem = 0;
    s = s + " ";
    for (auto x: s)
        if ('0' <= x && x <= '9')
        {
            dem++;
            p = p * 10 + (x - 48);
        }
}
```



```

else
{
    if (c[p] == 1) pmax = max(pmax,p);
    p = 0;
}
cout << dem << "\n" << pmax;
return 0;
}

```

### \* Chương trình minh họa bằng ngôn ngữ lập trình Python:

```

N = int(5e6)
def sangngto():
    c[0] = c[1] = 0;
    k = int(N ** 0.5)
    for i in range(2, k+1):
        if (c[i] == 1):
            for j in range(i*i, N+1,i): c[j] = 0
import sys
sys.stdin = open("ntmax.inp", "r")
sys.stdout = open("ntmax.out", "w")
c = [1] * (N + 3);
sangngto()
s = input()
s = s + ' ';
p = pmax = dem = 0;
for x in s:
    if ('0' <= x and x <= '9'):
        dem += 1
        p = p * 10 + ord(x) - 48;
    else:
        if (c[p]): pmax = max(pmax,p);
        p = 0;
print(dem)
print(pmax)

```

## BÀI 3. DÃY CON LIÊN TIẾP CÓ TỔNG CHIA HẾT CHO K

### \* Thuật toán đề xuất:

**Subtask1:** có 5/25 test tương ứng 1 điểm với  $n \leq 10^2$ ;

Duyệt  $(i, j)$  với  $1 \leq i \leq j \leq n$ :

- Tính tổng  $s = a[i] + a[i + 1] + \dots + a[j]$
- Nếu  $s$  chia hết cho  $k$  thì tăng biến đếm kết quả bài toán

Độ phức tạp:  $O(n^3)$

**Subtask2:** có 15/25 test tương ứng 3 điểm với  $n \leq 10^3$ ;

Cải tiến thuật toán trên bằng cách sử dụng mảng cộng dồn  $s[i] = a[1] + a[2] + \dots + a[i]$

Độ phức tạp:  $O(n^2)$

**Subtask3:** có 5/25 test tương ứng 1 điểm với  $n \leq 10^6$ .

Đặt  $s[i] = a[1] + a[2] + \dots + a[i]$

Xét tổng đoạn con:  $a[i] + a[i+1] + \dots + a[j] = s[j] - s[i-1]$  ( $1 \leq i \leq j \leq n$ )

Tổng này chia hết cho  $k$  tương đương với  $(s[j] - s[i-1])$  chia hết cho  $k$ , tương đương với  $s[j] \equiv s[i-1] \pmod k$  ( $s[j]$  đồng dư với  $s[i-1]$  theo modul  $k$ ).

Suy ra bài toán trở thành đếm cặp  $(i, j)$  sao cho  $s[i] \% k = s[j] \% k$  (với  $0 \leq i \leq j \leq n$ ).

Như vậy, có thể gán  $s[i] = s[i] \% k$ , và bài toán trở thành đếm số cặp phần tử bằng nhau trong dãy  $s[]$ .

Nếu tần số xuất hiện của  $s[i]$  trong dãy là  $x$  thì số cặp phần tử bằng nhau và bằng giá trị  $s[i]$  sẽ là  $x * (x - 1) / 2$ .

*Lưu ý cách xử lý:*

- Đối với ngôn ngữ lập trình C++: vì  $k \leq 10^9$  nên không thể dùng mảng để lưu tần số của các giá trị  $s[i]$ , mà có thể sử dụng **map** hoặc **unordered\_map**. Tuy nhiên, với  $n \leq 10^6$  sử dụng **map** sẽ bị lỗi thời gian, còn sử dụng **unordered\_map** thì themis báo lỗi biên dịch (vì trình biên dịch trong themis không hiểu **unordered\_map**). Vì vậy, cách xử lý thủ công là sắp xếp lại dãy  $s[]$  tăng dần, khi đó các phần tử bằng nhau sẽ ở gần nhau, và dễ dàng đếm được tần số của mỗi phần tử trong  $s[]$ .
- Đối với ngôn ngữ lập trình Python: cách xử lý khá đơn giản là sử dụng kiểu dữ liệu từ điển.

**Chủ đề: Duyệt + lý thuyết đồng dư**

**\* Chương trình minh họa bằng ngôn ngữ lập trình C++:**

```
#include <bits/stdc++.h>
#define N 1000000
#define ll long long
using namespace std;
ll a[N+3];
int n;
ll k, ans = 0;
int main()
{
    ios_base::sync_with_stdio(0);
    cin.tie(0); cout.tie(0);
```



```

freopen("chiak.inp","r",stdin);
freopen("chiak.out","w",stdout);
cin >> n >> k;
for (int x,i=1; i<=n; i++)
{
    cin >> x;
    a[i] = (a[i-1] + x + int(1e9) * k) % k;
}
sort(a+1,a+n+1);
a[n+1] = int(1e9+4);
int d = 1;
for (int i=1; i<=n+1; i++)
if (a[i] == a[i-1]) d++;
else
{
    ans += d * (d - 1)/2;
    d = 1;
}
cout << ans;
return 0;
}

```

---

**\* Chương trình minh họa bằng ngôn ngữ lập trình Python:**

---

```

import sys
sys.stdin = open("chiak.inp","r")
sys.stdout = open("chiak.out","w")
n, k = map(int,input().split())
a = [0] + list(map(int,input().split()))
for i in range(1,n+1):
    a[i] = (a[i-1] + a[i] + int(1e9) * k) % k;
dic = {}
ans = 0;
for x in a:
    if (dic.get(x) == None):
        dic[x] = 1
    else:
        ans += dic[x];
        dic[x] += 1
print(ans)

```

---

## BÀI 4. TẶNG QUÀ

**\* Thuật toán đề xuất:**

Để dàng nhận thấy bài 4 là bài dãy con tăng dài nhất với điều kiện tổng giá trị của các món quà là lớn nhất.

- **Subtask1:** có 10/30 test tương ứng 1 điểm với  $n \leq 10^3$ ;

Gọi  $dp[i]$  là tổng giá trị các gói quà lớn nhất kết thúc tại món quà thứ  $i$  (khi xét các món quà 1, 2, 3, ...,  $i$ ).

Bài toán cơ sở:  $dp[0] = 0$  – nếu không có món quà nào thì tổng giá trị các món quà bằng 0.

Xét món quà thứ  $i$ :  $1 \rightarrow n$

Xét món quà thứ  $j$ :  $0 \rightarrow i - 1$

Nếu  $a[j] < a[i]$  thì  $dp[i] = \max(dp[i], dp[j] + w[i])$ .

Độ phức tạp:  $O(n^2)$

- **Subtask2**: có 20/30 test tương ứng 3 điểm với  $n \leq 5 \cdot 10^5$ .

Thuật toán được cải tiến khi tìm món quà thứ  $j$  thật nhanh bằng cấu trúc cây chỉ số nhị phân - Fenwick Tree (hay còn gọi là cây BIT - bit index tree).

Lưu ý: vì  $a[i] \leq 10^9$  nên cần phải nén mảng  $a[i]$  xuống sao cho  $a[i] \leq 5 \cdot 10^5$  khi đó mới sử dụng cấu trúc cây Fenwick Tree.

Độ phức tạp:  $O(n \log n)$

**Chủ đề: Quy hoạch động + cấu trúc dữ liệu đặc biệt**

**\* Chương trình minh họa bằng ngôn ngữ lập trình C++:**

```
#include <bits/stdc++.h>
#define N int(5e5)
#define ll long long
using namespace std;
struct gift
{
    int a, w, id;
};
gift b[N+3];
ll T[N+3], dp[N+3], ans;
int n;
map <int,int> M;
bool cmp_label(gift X, gift Y)
{
    return X.a < Y.a;
}
bool cmp_id(gift X, gift Y)
{
    return X.id < Y.id;
}
ll get(int x)
{
    ll res = 0;
    while (x > 0)
    {
        res = max(res, T[x]);
        x = x - (x & (-x));
    }
    return res;
}
```



```

}

void update(int x, ll delta)
{
    while (x <= N)
    {
        T[x] = max(T[x], delta);
        x = x + (x & (-x));
    }
}

int main()
{
    ios_base::sync_with_stdio(0);
    cin.tie(0); cout.tie(0);
    freopen("qua.inp", "r", stdin);
    freopen("qua.out", "w", stdout);
    cin >> n;
    for (int i=1; i<=n; i++)
    {
        cin >> b[i].a >> b[i].w;
        b[i].id = i;
    }
    ///nen nhn b[i].a xuống để nhn có giá trị <= n
    sort(b+1, b+n+1, cmp_label);
    for (int i=1; i<=n; i++)
    if (M[b[i].a] == 0)
    {
        int x = M.size();
        M[b[i].a] = x;
        b[i].a = x;
    }
    else b[i].a = M[b[i].a];
    ///sau khi nen xong thi sap xep lai day quay ve trinh tu cu
    sort(b+1, b+n+1, cmp_id);
    ///tim ket qua
    for (int i=1; i<=n; i++)
    {
        ll x = get(b[i].a - 1);
        dp[i] = x + b[i].w;
        ans = max(ans, dp[i]);
        update(b[i].a, dp[i]);
    }
    cout << ans;
    return 0;
}

```