1 pA

照題目敘述把 mail 分類
策略: 按照題意建立兩棵 trie 最後走訪即可

```cpp
#include <bits/stdc++.h>
using namespace std;

#define CHARMAX 29

char s[32768][128];
int cnt;

struct Node {
    char s;
    vector<int> ans;
    Node *next[CHARMAX];
    void init(){
        for(int i = 0 ; i < CHARMAX ; i++)
            next[i] = 0;
    }
    Node(){init();}
    Node(char _s){
        init();
        s = _s;
    }
};

Node *root;
Node *other;
void insert(Node *r, char str[], int id){
    Node *now = r;
    for(int i = 0 ; str[i] ; i++){
        if(r == root){
            if(str[i] == '.') continue;
            if(str[i] == '+') break;
            if(str[i] == '@') break;
        }
        int next = (int)str[i];
        if(next == '.') next = 26;
        else if(next == '+') next = 27;
        else if(next == '@') next = 28;
        else if(next >= 'A' && next <= 'Z') next = next - 'A';
        else next = next - 'a';
        if(now->next[next] == NULL){
            now->next[next] = new Node(char(next));
        }
        now = now->next[next];
    }
    if((*now).ans.size()) cnt--;
    (*now).ans.push_back(id);
}
void travel(Node* now, bool p = 0){
    if( (*now).ans.size() ){
        if(p){
            printf("%d", (int)(*now).ans.size());
            for(int i = 0 ; i < (int)(*now).ans.size() ; i++){
                printf(" %s", s[(*now).ans[i]]);
            }
            printf("\n");
        }
```

```
57      }
58      for(int i = 0 ; i < CHARMAX ; i++){
59          if(now->next[i]){
60              travel(now->next[i], p);
61          }
62      }
63  }
64  bool check(char str[]){
65      int len = strlen(str);
66      char com[] = "@bmail.com";
67      char *com_now = com;
68      char *str_now = str + len - 10;
69      for(int i = 0 ; i < 10 ; i++, com_now++, str_now++){
70          if(*com_now == *str_now) continue;
71          if( ((*com_now) | 32) == ((*str_now) | 32) )continue;
72          return false;
73      }
74      return true;
75
76  }
77
78  int main(){
79      root = new Node();
80      other = new Node();
81      int n;
82      scanf("%d", &n);
83      cnt = n;
84      for(int i = 0 ; i < n ; i++){
85          scanf("%s", s[i]);
86          if(check(s[i])){
87              insert(root, s[i], i);
88          } else {
89              insert(other, s[i], i);
90          }
91      }
92      printf("%d\n", cnt);
93      travel(root, 1);
94      travel(other, 1);
95      return 0;
96  }
```

2 pB

把蛋糕擺好切下去問可以得到的最大單位體積以及切的長寬

策略： 先把長寬翻轉到同一個方向不難發現切下去的長或寬一定恰巧符合某個蛋糕的長或寬因此將寬大到小 sort 後枚舉每個長度只要高度大於當前枚舉的高度則將此蛋糕列入考慮同時維護寬度以及答案

```cpp
#include <bits/stdc++.h>
using namespace std;
#define f first
#define s second
typedef long long ll;
typedef pair<ll, ll> PLL;
bool cmp(const PLL &a, const PLL &b){
    return a.s > b.s;
}

int main(){
    int n;
    scanf("%d", &n);
    vector<PLL> v(n);
    for(int i = 0 ; i < n ; i++){
        scanf("%I64d%I64d", &v[i].f, &v[i].s);
        if(v[i].s > v[i].f) swap(v[i].f, v[i].s);
    }
    sort(v.begin(), v.end(), cmp);
    PLL ans;
    ll value = 0;
    for(int i = 0 ; i < n ; i++){
        int cnt = 0;
        for(int j = 0 ; j < n ; j++){
            if(v[j].f >= v[i].f)
                cnt++;
            if(v[j].s * v[i].f * cnt > value){
                ans = PLL(v[i].f, v[j].s);
                value = v[j].s * v[i].f * cnt;
            }
        }
    }
    printf("%I64d\n%I64d %I64d\n", value, ans.f, ans.s);
    return 0;
}
```

3 pD

給定 n 個人走一條線每個人有瞬間出現時間起點終點問每個人會和幾個人碰面

策略: 先將每個人存在的時間跟方向算出來對於方向一樣的他們重合時間的起點必須相同對於方向不同的則看他們重合時間是否有碰頭即可

```
1
2  #include <bits/stdc++.h>
3  using namespace std;
4  typedef pair<int, int> PII;
5  #define f first
6  #define s second
7
8  struct S{
9      int ts, te, s, e, d;
10     S(){}
11     S(int _t, int _s, int _e) : ts(_t), te(_t+abs(_e-_s)), s(_s), e(_e), d(e > s ? 1 : -1)
           {}
12     PII find(int start, int end){
13         int x = s + d * (start - ts);
14         int y = s + d * (end - ts);
15         return PII(min(x, y), max(x, y));
16     }
17 };
18
19
20 int main(){
21     int n;
22     while(~scanf("%d", &n)){
23         vector<S> v;
24         for(int i = 0 ; i < n ; i++){
25             int a, b, c;
26             scanf("%d%d%d", &a, &b, &c);
27             v.push_back(S(a, b, c));
28         }
29         for(int i = 0 ; i < n ; i++){
30             int ans = 0;
31             for(int j = 0 ; j < n ; j++){
32                 if(i == j) continue;
33                 S &x = v[i];
34                 S &y = v[j];
35                 if(x.te < y.ts || y.te < x.ts) continue;
36                 int start = max(x.ts, y.ts);
37                 int end = min(x.te, y.te);
38                 PII a = x.find(start, end);
39                 PII b = y.find(start, end);
40                 if(x.d == y.d){
41                     if(a.f == b.f) ans++;
42                 } else if(x.d == -1){
43                     if(a.s >= b.f && a.f <= b.s) ans++;
44                 } else {
45                     if(b.s >= a.f && b.f <= a.s) ans++;
46                 }
47             }
48             printf("%d\n", ans);
49         }
50     }
51
52     return 0;
53 }
```

4 pF

給定一堆東西存在的時間現在要分配時間給他們規定每個東西要有相同的時間問總分配時間為多少
策略: 二分搜答案再用 greedy 檢查快截止的先給

```cpp
#include <bits/stdc++.h>
using namespace std;
#define f first
#define s second
typedef pair<int, int> PII;
vector<PII> v;
bool check(int s){
    int remain[(int)v.size()];
    fill(remain, remain+v.size(), s);
    int id = 0, now = 0;
    priority_queue< PII, vector<PII>, greater<PII> > PQ;
    while(!(id == (int)v.size() && PQ.empty())){
        if(PQ.empty())
            now = v[id].f;
        for( ; id < (int)v.size() && v[id].f == now ; id++)
            PQ.push(PII(v[id].s, id));
        int next_now = min(PQ.top().f, id == (int)v.size() ? 0x3f3f3f3f : v[id].f);
        int quota = next_now - now;
        now = next_now;

        while(quota && PQ.size()){
            int u = PQ.top().s;
            int value = min(remain[u], quota);
            quota -= value;
            remain[u] -= value;
            if(remain[u] == 0) PQ.pop();
        }
        if(PQ.size() && PQ.top().f <= now){
            return false;
        }
    }
    return true;
}
bool cmp(const PII &a, const PII &b){
    return a.f < b.f;
}
int main(){
    int n, s = 0, step = 16384, l, r;
    scanf("%d", &n);

    for(int i = 0 ; i < n ; i++){
        scanf("%d%d", &l, &r);
        v.push_back(PII(l, r));
        step = min(step, r -l);
    }
    sort(v.begin(), v.end(), cmp);
    while(step){
        if(check(s+step)){
            s += step;
        } else {
            step >>= 1;
        }
    }
    printf("%d\n", s * (int)v.size());
    return 0;
}
```

5 pI

現在有 n 顆球 k 種顏色每顆球事先都有了顏色問你最少要重塗幾顆球才可以讓每種顏色一樣多
策略: 先算出每種球要幾顆接著如果有一種球超出了平均則代表他要被塗成其他顏色

```cpp
#include <bits/stdc++.h>
using namespace std;

int cnt[128];
int main(){
    int n, m, t;
    scanf("%d%d", &n, &m);
    int avg = n / m, ans = 0;
    for(int i = 0 ; i < n ; i++){
        scanf("%d", &t);
        if(cnt[t] < avg) cnt[t]++;
        else ans++;
    }
    printf("%d\n", ans);
    return 0;
}
```

6 pJ

給定二維平面地圖撞到東西就換方向走 (URDL) 問可以走幾個格子

策略: 開個 3 為表格 dp[h][w][d] 來記錄如果走過就代表會形成迴圈了跳出並計算總共可以走到幾個地方

```cpp
#include <bits/stdc++.h>
using namespace std;

char t[12][12];
int d[4][2] = {{-1, 0}, {0, 1}, {1, 0}, {0, -1}};
int n, m;
bool dp[12][12][4];
int main(){
//    freopen("pj.in", "r", stdin);
    scanf("%d%d", &n, &m);
    int x, y, dir;
    for(int i = 0 ; i < n ; i++){
        scanf("%s", t[i]);
        for(int j = 0 ; j < m ; j++){
            if(t[i][j] != '.' && t[i][j] != '*'){
                x = i, y = j;
                switch(t[i][j]){
                    case 'U': dir = 0; break;
                    case 'R': dir = 1; break;
                    case 'D': dir = 2; break;
                    case 'L': dir = 3; break;
                }
            }
        }
    }
    int ans = 0;
    bool flag = 1;
    while(flag){
        if(dp[x][y][dir]) break;
        flag = 0;
        t[x][y] = '-';
        dp[x][y][dir] = 1;
        for(int i = 0 ; i < 4 ; i++){
            int next_dir = (dir+i) % 4;
            int nx = x + d[next_dir][0];
            int ny = y + d[next_dir][1];
            if(nx < 0 || ny < 0 || nx >= n || ny >= m) continue;
            if(t[nx][ny] == '*')continue;
            flag = 1;
            x = nx;
            y = ny;
            dir = next_dir;
            break;
        }
    }
    for(int i = 0 ; i < n ; i++)
        for(int j = 0 ; j < m ; j++)
            ans += t[i][j] == '-';
    printf("%d\n", ans);
    return 0;
}
```