实现技巧

主要介绍acm竞赛中常见的编程技巧

1.代码规范与个人风格

统一规范的编程风格可以提高代码的可阅读性

- 缩进

```
int main()
{
    int a, b, c;
    if(...)
        if(...)
        else
    else
}
```

- 花括号

```
    if() {
        ...
    }else {
        ...
    }
```

- 适当的空格

```
int f = (a + b) * c;
```

- 命名规范

  变量名或函数名应尽量与用途相联系，切忌使用相似的变量名，如

```
//warning
int t,tt;
vector<int> v[N],vv;
```

更多规范可以参考：https://oi.men.ci/code-style-oi/?tdsourcetag=s_pctim_aiomsg

2.头文件

适当的头文件可以加快编程的速度

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
typedef double db;
typedef pair<int,int> pii;
```

```
typedef vector<int> vi;
#define dd(x) cout << #x << "=" << x << ","
#define de(x) cout << #x << "=" << x << endl
#define rep(i,a,b) for(int i=(a);i<(b);++i)
#define per(i,a,b) for(int i=(b-1);i>=a;--i)
#define all(x) (x).begin(),(x).end()
#define sz(x) (int)(x).size()
#define mp make_pair
#define pb push_back
#define fi first
#define se second
const int N = 101010;
```

3.使用c++标准库和命名空间

使用前人造好的轮子

```
    // swap
    int u = 0, v = 1;
    std::swap(u , v); // swap
    set<int> A , B;
    std::swap(A , B); // O(1)

    // minimal & maximal
    int a[20] , n = 20;
    rep(i,0,n) a[i] = i;
    cout << *std::max_element(a , a + n) << endl;// [a , a+n)
    cout << *std::min_element(a , a + n) << endl;

    // discretization
    vi V;// about 10 int
    sort(all(V));V.erase(unique(all(V)),V.end());
    #define rk(x) upper_bound(all(V) , x) - V.begin()

    // sort
    int cnt[20];
    sort(all(V),[&](int a,int b){return cnt[a]<cnt[b];}); // c++11
    vector<vi> Vv;
    sort(all(Vv));

    // sort with id
    vector<pii> p; // pair
    rep(i,0,20) p.pb(mp(rand(),i));
    sort(all(p));

    // permutation
    rep(i,0,7) a[i] = i;
    do{
    // Cal;
    }while(next_permutation(a , a + 7));
```

更多内容可以访问c++参考手册：

https://zh.cppreference.com/w/cpp

http://www.cplusplus.com/reference/

4.位运算

利用计算机二进制存储的特性

```
/*  &   |   ^    >>   <<
    与  或  异或  右移  左移 */

    // __builtin_popcount()
    int cnt1[1<<6];
    rep(i,1,1<<6) cnt1[i] = cnt1[i >> 1] + (i & 1);

    #define lowbit(x) x&(-x)
    // true while x == 2^k
    bool judge2k(int x) {
        return x == lowbit(x);
    }
    // binary representation
    for(int d = 10;d>=0;d--) printf("%c","01"[mask>>d&1]);

    // deal with subsets
    rep(mask,0,1<<10)
        for(int j=mask;j;j=(j-1)&mask)
          ;// cal(j)

    // high-dimensional prefix-sum*
    int f[1<<10];
    rep(i,0,10) rep(j,0,1<<10) if(j>>i&1) pp(f[j],f[j^(1<<i)]);
```

> $*f[i] = \sum_{i|j=i} f[j]$

5.运算技巧

模意义下的运算

```
const int MOD = 1e9 + 7;
// add-mod
void pp(int &x,int d){ if((x+=d)>=MOD) x-=MOD;}
// minus-mod -> pp(a, MOD - x);
// multiply-mod
int mul(int a,int b){ return ll(a)*b%MOD;}
//fast-pow
int Pow(ll x,ll k) {ll r=1;for(;k;k>>=1,x=mul(x,x)) if(k&1) r=mul(r,x); return
r;}
// inversion
int inverse(int x,int p) {return Pow(x,p-2,p);} // p should be prime
int main(){
    // ternary operator
    int C[10][10] = {{1}};
    rep(i,1,10) rep(j,0,i+1) C[i][j] = j ? (C[i-1][j-1] + C[i-1][j]) : 1;
    //__gcd()
    int gcd(int a,int b) {return b ? gcd(b, a%b) : a};
    // reference
    int &r=f[10];
    rep(i,0,10) r+=i;
    return 0;
```

```
}
```

6. 图

非线性逻辑结构

```
// dsu
int fa[N];
void I(int n) { rep(i,0,n) fa[i] = i;}
int F(int x){ return fa[x] == x ? x : fa[x] = F(fa[x]);}
void M(int x,int y){ fa[F(x)] = F(y);}

// tree-dp
int sz[N];
void dfs(int c,int par){
    sz[c] = 1;
    for(auto t : g[c]) if(t != par) { // c++11
        dfs(t , c);
        sz[c] += sz[t];
    }
}
int main()
{
    rep(i,0,m) {
        int u,v;scanf("%d%d",&u,&v);
        g[u].pb(v);
        g[v].pb(u); //undirected graph
    }
}
```

7. 赋值

```
// fill function
    memset(a,0,sizeof(a));
    memset(a,0,sizeof(a[0])*20);
    std::fill(a, a + 20, 0);// fill any number
    std::fill_n(a, 20, 0);
    vi e[N];
    std::fill_n(e, N, vi());
```