随机模板

```
一、rand()随机数
范围: 0~32767
#include <iostream>
#include <ctime>
using namespace std;
int main()
{
  srand(unsigned(time(0)));
  int count = rand() % 3 + 1;    //范围1~3
  int count1 = rand() % 3;    //范围0~2
  cout << count << endl << count1 << endl;
  return 0;
}


二、mt19937随机数
范围: 无限制, 但是可以自己设定。
C++(无范围)
#include <iostream>
#include <chrono>
#include <random>
using namespace std;
int main()
{
    // 随机数种子
  unsigned seed = std::chrono::system_clock::now().time_since_epoch().count();
    mt19937 rand_num(seed);  // 大随机数
  cout << rand_num() << endl;
  return 0;
}

C++(手动加上范围)
#include <iostream>
#include <chrono>
#include <random>
using namespace std;
int main()
{
  // 随机数种子
  unsigned seed = std::chrono::system_clock::now().time_since_epoch().count();
  mt19937 rand_num(seed);  // 大随机数
  uniform_int_distribution<long long> dist(0, 1000000000);  // 给定范围
  cout << dist(rand_num) << endl;
  return 0;
}
```

```
三、[0,x-1]
mt19937 mrand(random_device{}());
int rnd(int x) { return mrand() % x;}


ll rand_int(ll l, ll r) //[l, r]
{
  static mt19937_64 gen(chrono::steady_clock::now().time_since_epoch().count());
  return uniform_int_distribution<ll>(l, r)(gen);
}
```

高精度模板（仅正数）

```
const int maxn=1000;
struct bign{
    int d[maxn], len;

    void clean() { while(len > 1 && !d[len-1]) len--; }

    bign()              { memset(d, 0, sizeof(d)); len = 1; }
    bign(int num)      { *this = num; }
    bign(char* num) { *this = num; }
    bign operator = (const char* num){
        memset(d, 0, sizeof(d)); len = strlen(num);
        for(int i = 0; i < len; i++) d[i] = num[len-1-i] - '0';
        clean();
        return *this;
    }
    bign operator = (int num){
        char s[20]; sprintf(s, "%d", num);
        *this = s;
        return *this;
    }

    bign operator + (const bign& b){
        bign c = *this; int i;
        for (i = 0; i < b.len; i++){
            c.d[i] += b.d[i];
            if (c.d[i] > 9) c.d[i]%=10, c.d[i+1]++;
        }
        while (c.d[i] > 9) c.d[i++]%=10, c.d[i]++;
        c.len = max(len, b.len);
        if (c.d[i] && c.len <= i) c.len = i+1;
        return c;
    }
```

```cpp
    bign operator - (const bign& b){
        bign c = *this; int i;
        for (i = 0; i < b.len; i++){
            c.d[i] -= b.d[i];
            if (c.d[i] < 0) c.d[i]+=10, c.d[i+1]--;
        }
        while (c.d[i] < 0) c.d[i++]+=10, c.d[i]--;
        c.clean();
        return c;
    }
    bign operator * (const bign& b)const{
        int i, j; bign c; c.len = len + b.len;
        for(j = 0; j < b.len; j++) for(i = 0; i < len; i++)
            c.d[i+j] += d[i] * b.d[j];
        for(i = 0; i < c.len-1; i++)
            c.d[i+1] += c.d[i]/10, c.d[i] %= 10;
        c.clean();
        return c;
    }
    bign operator / (const bign& b){
        int i, j;
        bign c = *this, a = 0;
        for (i = len - 1; i >= 0; i--)
        {
            a = a*10 + d[i];
            for (j = 0; j < 10; j++) if (a < b*(j+1)) break;
            c.d[i] = j;
            a = a - b*j;
        }
        c.clean();
        return c;
    }
    bign operator % (const bign& b){
        int i, j;
        bign a = 0;
        for (i = len - 1; i >= 0; i--)
        {
            a = a*10 + d[i];
            for (j = 0; j < 10; j++) if (a < b*(j+1)) break;
            a = a - b*j;
        }
        return a;
    }
    bign operator += (const bign& b){
        *this = *this + b;
        return *this;
    }

    bool operator <(const bign& b) const{
        if(len != b.len) return len < b.len;
        for(int i = len-1; i >= 0; i--)
            if(d[i] != b.d[i]) return d[i] < b.d[i];
        return false;
    }
    bool operator >(const bign& b) const{return b < *this;}
    bool operator<=(const bign& b) const{return !(b < *this);}
    bool operator>=(const bign& b) const{return !(*this < b);}
    bool operator!=(const bign& b) const{return b < *this || *this < b;}
    bool operator==(const bign& b) const{return !(b < *this) && !(b > *this);}

    string str() const{
        char s[maxn]={};
        for(int i = 0; i < len; i++) s[len-1-i] = d[i]+'0';
        return s;
    }
};

istream& operator >> (istream& in, bign& x)
{
    string s;
    in >> s;
    x = s.c_str();
    return in;
}

ostream& operator << (ostream& out, const bign& x)
{
    out << x.str();
    return out;
}
```

莫队

```
#include<bits/stdc++.h>
using namespace std;
using ll=long long;
#define int ll
#define ull unsigned long long
#define pii pair<int,int>
#define vc vector
#define vi vector<int>
#define db double
#define PI acos(-1.0)
/* #define ls u<<1 */
/* #define rs u<<1|1 */
#define mk make_pair
#define fi first
```

```cpp
#define se second
#define forn(i, n) for (int i = 1; i <= n; ++i)
#define forr(i, n) for (int i = n; i >= 1; --i)
#define IOFast() ios::sync_with_stdio(0),cin.tie(0),cout.tie(0)
#ifndef ONLINE_JUDGE
#define dbg(x...) do { cout << "\033[33;1m " << #x << " -> "; err(x); } while (0)
void err() { cout << "\033[39;0m" << endl; }
template<template<typename...> class T, typename t, typename... A>
void err(T<t> a, A... x) { for (auto v: a) cout << v << ' '; err(x...); }
template<typename T, typename... A>
void err(T a, A... x) { cout << a << ' '; err(x...); }
#else
#define dbg(...)
#endif

#ifndef ONLINE_JUDGE
#define fileopen() do{ freopen("in", "r", stdin); freopen("out", "w", stdout);  } while
(0)
#else
#define fileopen()
#endif

int n, q;
int block;
const int maxn = 2e6 + 10;
int a[maxn];
int cnt[maxn];
int res[maxn];
int ans = 0;

struct node{
  int l, r, id;
  bool operator < (node cmp) const{
    if((l / block) == (cmp.l / block)){
      if((l / block) % 2){
        return r < cmp.r;
      }else{
        return r > cmp.r;
      }
    }else{
      return l < cmp.l;
    }
  }
}t[maxn];

void add(int x){
  cnt[x] ++ ;
  if(cnt[x] == 1)
    ans ++ ;
}

void del(int x){
  cnt[x] -- ;
  if(!cnt[x])
    ans -- ;
}

signed main(){
  IOFast();
  cin >> n;
  block = sqrt(1.0*n);
  forn(i, n){
    cin >> a[i];
  }
  cin >> q;
  forn(i, q){
    cin >> t[i].l >> t[i].r;
    t[i].id = i;
  }
  sort(t + 1, t + 1 + q);
  int l = t[1].l, r = t[1].l - 1;
  ans=0;
  forn(i, q){
    while(l > t[i].l) add(a[ -- l]);
    while(r < t[i].r) add(a[ ++ r]);
    while(l < t[i].l) del(a[l ++ ]);
    while(r > t[i].r) del(a[r -- ]);
    res[t[i].id] = ans;
  }
  for(int i = 1; i <= q;  ++ i){
    cout << res[i] << '\n';
  }
}
```

模拟退火

```cpp
#include <bits/stdc++.h>
#define down 0.996//徐徐降温

using namespace std;

int n;
struct node{
int x;
int y;
int w;
}object[2005];//存下物体的坐标
```

```
double ansx,ansy,answ;//最终答案
double energy(double x,double y)//根据物理学知识,能量总和越小越稳定
{
   double r=0,dx,dy;
   for (int a=1;a<=n;a++)
   {
      dx=x-object[a].x;
      dy=y-object[a].y;
      r+=sqrt(dx*dx+dy*dy)*object[a].w;
   }
      return r;
}
void sa()//模拟退火
{
   double t=3000;//温度要足够高
   while (t>1e-15)//略大于0
   {
      double ex=ansx+(rand()*2-RAND_MAX)*t;//随机产生新的答案
      double ey=ansy+(rand()*2-RAND_MAX)*t;
      double ew=energy(ex,ey);
      double de=ew-answ;
      if (de<0)//如果此答案更优，就接受
      {
         ansx=ex;
         ansy=ey;
         answ=ew;
      }
      else if(exp(-de/t)*RAND_MAX>rand())//否则根据多项式概率接受
      {
         ansx=ex;
         ansy=ey;
      }
      t*=down;
   }
}
void solve()//多跑几遍退火,增加得到最优解的概率
{
   sa();
   sa();
   sa();
   sa();
}
int main() {
cin>>n;
for (int a=1;a<=n;a++)
{
   scanf("%d%d%d",&object[a].x,&object[a].y,&object[a].w);
   ansx+=object[a].x;
   ansy+=object[a].y;
}
ansx/=n;//以平均数作为初始答案
ansy/=n;
answ=energy(ansx,ansy);
solve();
printf("%.3lf %.3lf\n",ansx,ansy);//华丽的输出
    return 0;
}
```

二分

写法一：

```
while(l<=r)
{
  int mid=(l+r)/2;
  if(chk(mid))
  {
    res=mid;
    l=mid+1;
  }
  else
  {
    r=mid-1;
  }
}
```

写法二：

```
while(l<r)
{
  int mid=(l+r)/2;
  if(chk(mid))
    l=mid;
  else r=mid-1;
}
```

求长度大于x的区间平均最大值

```
#include<iostream>
#include<cstdio>
#include<string>
#include<vector>
#include<cmath>
#include<stack>
```

```cpp
#include<queue>
#include<map>
#include<unordered_map>
#include<set>
#include<cstring>
#include<algorithm>
#include<climits>
#include<numeric>
#include<cassert>
#include<iomanip>
#define int long long
#define pii pair<int,int>
#define forn(i,t) for(int i=1;i<=t;i++)
#define forr(i,t) for(int i=t;i>=1;i--)
#define IOFast() ios::sync_with_stdio(0),cin.tie(0),cout.tie(0)
using namespace std;
const int maxn=1e5+10;
const double eps=1e-9;
int n,m,x,y;
vector<int> a,b;
double sum[maxn];
void init(){
  cin>>n>>m>>x>>y;
  a.resize(n+1);
  b.resize(m+1);
  forn(i,n)
    cin>>a[i];
  forn(i,m)
    cin>>b[i];
}
bool chk(double tar,const vector<int> & v,int pos){
  int len=v.size();
  double mn=0;
    forn(i,len-1)
    sum[i]=sum[i-1]+v[i]-tar;
    for(int i=pos;i<len;i++)
    {
        mn=min(mn,sum[i-pos]);
        if(sum[i]>=mn) return true;
    }
    return false;
}
void solve(){
  double res=0;
  double l=0,r=1e5;
  while(abs(l-r)>eps)
  {
    double mid=(l+r)/2.0;
    if(chk(mid,a,x))
```

```
      l=mid;
    else r=mid;
  }
    cout<<l<<endl;
  res+=l;
  l=0;r=1e5;
  while(abs(l-r)>eps)
  {
    double mid=(l+r)/2.0;
    if(chk(mid,b,y))
      l=mid;
    else r=mid;
  }
  res+=l;
  cout<<fixed<<setprecision(10)<<res<<'\n';
}
signed main(){
  IOFast();
  init();
  solve();
}
```

三分

```
//凹函数
 ll l1=0,r1=l,mid,mmid;

        while(l1<r1-2)
        {
            mid=(l1+r1)/2;
            mmid=(mid+r1)/2;

            if(fun(mid)>fun(mmid))
                l1=mid;
            else r1=mmid;
        }
    mid=(l1+r1)/2;
    printf("%lld\n",min(fun(mid),min(fun(l1),fun(r1))));
//凸函数
db mid, mmid;
  while(l < r - 1e-8){
    db k=(r-l)/3.0;
    mid = l+k;
    mmid = r-k;
    if(func(mid) > func(mmid))
```

```
        r = mmid;
      else l = mid;
    }
    mid = (l + r) / 2;
    cout << fixed << setprecision(5)  << mid << '\n';
```

三分套三分

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

using namespace std;

const int N = 110;
const double eps = 1e-3;
struct Point{
    double x, y, z;
    Point () {x = y = z = 0; }
    Point (double _x, double _y, double _z) {
        x = _x; y = _y; z = _z;
    }
    Point operator + (const Point& p) {
        return Point(x + p.x, y + p.y, z + p.z);
    }
    Point operator - (const Point& p) {
        return Point(x - p.x, y - p.y, z - p.z);
    }
    double len() {
        return sqrt(x * x + y * y + z * z);
    }
}p[N];
int n;

double cal3(double x, double y, double z) {
    double res = 0;
    for (int i = 1; i <= n; i++) {
        res = max(res, (p[i] - Point(x, y, z)).len());
    }
    return res;
}

double cal2(double x, double y) { //三分第三维
    double res = 1e18;
    double l = -100000.0, r = 100000.0;
    while (r - l > eps) {
        double m1 = l + (r - l) / 3.0;
        double m2 = l + (r - l) / 3.0 * 2.0;
        double res1 = cal3(x, y, m1), res2 = cal3(x, y, m2);
        res = min(res, min(res1, res2));
        if (res1 < res2) r = m2;
        else l = m1;
    }
    return res;
}

double cal1(double x) {    //三分二维
    double res = 1e18;
    double l = -100000.0, r = 100000.0;
    while (r - l > eps) {
        double m1 = l + (r - l) / 3.0;
        double m2 = l + (r - l) / 3.0 * 2.0;
        double res1 = cal2(x, m1), res2 = cal2(x, m2);
        res = min(res, min(res1, res2));
        if (res1 < res2) r = m2;
        else l = m1;
    }
    return res;
}

int main () {
    scanf("%d", &n);
    for (int i = 1; i <= n; i++)
        scanf("%lf%lf%lf", &p[i].x, &p[i].y, &p[i].z);
    double res = 1e18;
    double l = -100000.0, r = 100000.0;
    while (r - l > eps) {    //三分第一维
        double m1 = l + (r - l) / 3.0;
        double m2 = l + (r - l) / 3.0 * 2.0;
        double res1 = cal1(m1), res2 = cal1(m2);
        res = min(res, min(res1, res2));
        if (res1 < res2) r = m2;
        else l = m1;
    }
    printf("%.15f\n", res);
    return 0;
}
```

计数排序

```
#include <iostream>
using namespace std;
const int MAXN = 100000;
const int k = 1000; // range(范围)
int a[MAXN], c[MAXN], ranked[MAXN];

int main() {
    int n;
    cin >> n;
    for (int i = 0; i < n; ++i) {
        cin >> a[i];
        ++c[a[i]];
    }
    for (int i = 1; i < k; ++i)
        c[i] += c[i-1];
    for (int i = n-1; i >= 0; --i)
        ranked[--c[a[i]]] = a[i];//如果是i表达的是原数标号，a[i]就是排序后的正确序列
    for (int i = 0; i < n; ++i)
        cout << ranked[i] << endl;
    return 0;
}
```