
A

暴力

略。

B

> *如果$f(l)*f(r)<0$那么$f(x)=0$，x必定在(l,r)内！*

好一道神仙题啊！！

- 我们设$b(i)=a(i)-a(i+\frac{n}{2})$
- 那么显然 $b(i)=-b(i+\frac{n}{2})$

由于$|a(i)-a(i+1)|\leq 1\rightarrow|b(i)-b(i+1)|\leq 2\rightarrow b()$具有相同的奇偶性

那么我们每次就只需要二分啦~

C

> *。。。缺个脑子。。*

zhouzhendong

神奇的构造题！

- 首先我们先满足集合内的点两两之间没有连边
- 再次基础上我们每次都选入度为0的点即可

D

> *每次我们枚举一条线段作为底边 寻找与之相匹配的点*
>
> *实际上就相当于 以这条线段为y轴 进行坐标轴旋转*
>
> *我们只需要快速维护出旋转过后的坐标轴下 每个点按x这一维排序的顺序即可*
>
> *（怎么维护？请看下文*

反证一下

- 以斜率最小的那条线为y轴 x的顺序其实就是以原来y轴为顺序。。（反之则这条直线不是斜率最小的！
- 每次枚举到下一条边 x的顺序改变的只会是上一条边的两个端点（反之则这两条线之间还有线。。

E

> *其实我们可以把$a_i * x + b_i$看作一条向量(a_i, b_i)*

> *再看一下这个不等式哈～*
>
> $a_i * x + b_i < a_j * x + b_j (假设 a_i < a_j)$
>
> $$\Longrightarrow x \leq -\frac{b_i - b_j}{a_i - a_j}$$
>
> *所以我们实际上是要维护一个上秃(凸)折线啊！*
>
> *再用个边分治或者点分治就ok啦！*
>
> *（才没有OK）反正就是那种做到你怀疑人生的那种数据结构题。。*

边分治的话还要处理一下菊花图。。。（不过这种应该算是常规操作（虽然我也就补这题的时候才会的。。

Code

A

```
#include <bits/stdc++.h>
#define ll long long
using namespace std;
const int N = 3010;
ll qz[N][N];
int a[N][N], len[N], now[N], all[N], num[N];

int main() {
```

```
  int n, m;
  scanf("%d%d", &n, &m);
  for(int i = 1, x, y; i <= n; ++i) {
    scanf("%d%d", &x, &y);
    all[i] = y;
    a[x][++len[x]] = y;
  }
  sort(all + 1, all + 1 + n);
  int tot = unique(all + 1, all + 1 + n) - all - 1;
  for(int i = 2; i <= m; ++i) {
    if(!len[i]) continue;
    sort(a[i] + 1, a[i] + 1 + len[i]);
    for(int j = 1; j <= len[i]; ++j) {
      qz[i][j] = qz[i][j - 1] + a[i][j];
      a[i][j] = lower_bound(all + 1, all + 1 + tot,
a[i][j]) - all;
      num[a[i][j]]++;
    }
  }
  ll ans = 1e18;
  for(int i = 1; i <= m; ++i) now[i] = 1;
  for(int i = n; i >= 0; --i) {
    ll pos = 0;
    int sum = 0, maxn = 0;
    for(int j = 2; j <= m; ++j) {
      if(len[j] <= i) {
        sum += len[j];
        maxn = max(maxn, len[j]);
        continue;
      } else {
        maxn = max(maxn, i);
```

```
        sum += i;
        pos += qz[j][len[j] - i];
        num[a[j][now[j]]]--;
        now[j]++;
      }
    }
    if(n - sum <= maxn) {
      int tmp = maxn - (n - sum) + 1;
      // printf("pos=%lld maxn=%d sum=%d\n",
pos,maxn,sum);
      for(int j = 1; j <= tot; ++j) {
        if(num[j] <= tmp) {
          tmp -= num[j];
          pos += 1ll * num[j] * all[j];
        } else {
          pos += 1ll * tmp * all[j];
          tmp = 0;
        }
        if(!tmp) break;
      }
      if(tmp) continue;
    }
    ans = min(ans, pos);
  }
  printf("%lld\n", ans);
  return 0;
}
```

B

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

int ask(int x) {
  if(x > n) x -= n;
  printf("? %d\n", x);
  fflush(stdout);
  int y;
  scanf("%d", &y);
  return y;
}

int get(int x) {
  if(x <= n / 2) return ask(x) - ask(x + n / 2);
  else return ask(x) - ask(x - n / 2);
}

void print(int x) {
  if(x > n / 2) x -= n / 2;
  printf("! %d\n", x);
  fflush(stdout);
  exit(0);
}

int main() {
  scanf("%d", &n);
  int l = 0, r = n / 2, vl = ask(n) - ask(n / 2), vr
= -vl;
  if(vl == 0) print(n / 2);
  else if(vl > 0) {
```

```
    if(vl & 1) print(-1);
  } else {
    if(vr & 1) print(-1);
  }
  while(l <= r) {
    int mid = (l + r) >> 1;
    int pos = get(mid);
    if(!pos) print(mid);
    if(1ll * pos * vl < 0) r = mid - 1;
    else l = mid + 1;
  }
  print(-1);
  return 0;
}
```

C

```
#include <bits/stdc++.h>
using namespace std;
const int N = 1e6 + 10;
struct data {
  int nt, to;
} a[N];
int head[N], vis[N], d[N], c[N], b[N], g[N], cnt = 0;

void add(int x, int y) {
  a[++cnt].to = y;
  a[cnt].nt = head[x];
  head[x] = cnt;
}
```

```
void dfs(int x) {
  for(int i = head[x]; i; i = a[i].nt) {
    int to = a[i].to;
    if(vis[to]) {
      continue;
    }
    vis[to] = 1;
    g[++g[0]] = to;
  }
}

int main() {
  int n, m;
  scanf("%d%d", &n, &m);
  for(int i = 1, x, y; i <= m; ++i) {
    scanf("%d%d", &x, &y);
    d[y]++;
    add(x, y);
  }
  for(int i = 1; i <= n; ++i) {
    if(!d[i]) c[++c[0]] = i, b[++b[0]] = i;
  }
  if(!c[0]) {
    c[++c[0]] = b[++b[0]] = 1;
    vis[1] = 1;
  }
  for(;;) {
    for(int j = 1; j <= 2; ++j) {
      g[0] = 0;
      for(int i = 1; i <= b[0]; ++i) {
```

```
        dfs(b[i]);
      }
      for(int i = 0; i <= g[0]; ++i) {
        b[i] = g[i];
      }
    }
    if(!b[0]) break;
    for(int i = 1; i <= b[0];++i) {
      c[++c[0]] = b[i];
    }
  }
  printf("%d\n", c[0]);
  for(int i = 1; i <= c[0]; ++i) {
    printf("%d ", c[i]);
  }
  return 0;
}
```

D

```
#include <bits/stdc++.h>
#define ll long long
using namespace std;
const int N = 2010;
struct data {
  ll x, y;
  data() { x = y = 0; }
  data(ll _x, ll _y) { x = _x, y = _y; }
  data operator - (data C) { return data(x - C.x, y -
C.y); }
```

```
} a[N];
struct line {
  int x, y;
  data p;
} b[N * N];
int rk[N], id[N];

bool cmp(data A, data B) {
  return A.x == B.x ? A.y < B.y : A.x < B.x;
}

ll cross(data A, data B) {
  return A.x * B.y - A.y * B.x;
}

bool comp(line A, line B) {
  return cross(A.p, B.p) > 0;
}

ll Abs(ll x) {
  if(x < 0) x = -x;
  return x;
}

ll S;

int main() {
  int n;
  scanf("%d%lld", &n, &S);
  S *= 2ll;
  for(int i = 1; i <= n; ++i) {
```

```
    scanf("%lld%lld", &a[i].x, &a[i].y);
  }
  sort(a + 1, a + 1 + n, cmp);
  for(int i = 1; i <= n; ++i) {
    id[i] = rk[i] = i;
  }
  int m = 0;
  for(int i = 1; i <= n; ++i)
    for(int j = i + 1; j <= n; ++j) {
      b[++m].x = i, b[m].y = j;
      b[m].p = a[j] - a[i];
    }
  sort(b + 1, b + 1 + m, comp);
  for(int i = 1; i <= m; ++i) {
    int zd = b[i].x, yd = b[i].y;
    if(rk[zd] > rk[yd]) swap(zd, yd);
    int l = 1, r = rk[zd] - 1;
    while(l <= r) {
      int mid = (l + r) >> 1;
      ll Area = Abs(cross(b[i].p, a[id[mid]] -
a[zd]));
      if(Area == S) {
        printf("Yes\n");
        printf("%lld %lld\n", a[zd].x, a[zd].y);
        printf("%lld %lld\n", a[yd].x, a[yd].y);
        printf("%lld %lld\n", a[id[mid]].x,
a[id[mid]].y);
        return 0;
      } else if(Area > S) l = mid + 1;
      else r = mid - 1;
    }
```

```
    swap(rk[zd], rk[yd]);
    swap(id[rk[zd]], id[rk[yd]]);
  }
  puts("No");
  return 0;
}
```

E

```
#include <bits/stdc++.h>
#define ll long long
using namespace std;
const int N = 2e5 + 10;

struct P {
  ll x, y;

  P() { x = y = 0; }
  P(ll _x, ll _y) { x = _x, y = _y; }

  P operator + (P C) { return P(x + C.x, y + C.y); }
  P operator - (P C) { return P(x - C.x, y - C.y); }

  ll operator * (P C) {
    if(!x && !C.x) return 0ll;
    if(!x) return -y;
    if(!C.x) return C.y;
    if(x * C.y == y * C.x) return 0ll;
    if(1.0 * y / x < 1.0 * C.y / C.x) return 1;
    else return -1;
```

```
  }

  bool operator < (P &C) const {
    return x == C.x ? y < C.y : x < C.x;
  }
} pa[N * 20], ans[N * 20], qa[N * 10], qb[N * 10];

struct E {
  int head[N], Next[N << 1], to[N << 1], tot;
  P w[N << 1];

  E() { tot = 1; }

  void add(int x, int y, P p) {
    to[++tot] = y;
    w[tot] = p;
    Next[tot] = head[x];
    head[x] = tot;
  }

  void adde(int x, int y, P p) {
    add(x, y, p), add(y, x, p);
  }
} S, T;

int fr[N], us[N], sz[N];
int n, m, now, na, nb, mx, bu, bv, bid, tot, len,
cnt;

void build(int x, int fa) {
  for(int i = S.head[x]; i; i = S.Next[i]) {
```

```
    int to = S.to[i];
    if(to == fa) {
      continue;
    }
    T.adde(fr[x], ++now, P(0, 0));
    T.adde(now, to, S.w[i]);
    fr[x] = now;
    build(to, x);
  }
}

void gotsz(int x, int fa) {
  sz[x] = 1;
  for(int i = T.head[x]; i; i = T.Next[i]) {
    int to = T.to[i];
    if(us[i >> 1] || to == fa) {
      continue;
    }
    gotsz(to, x);
    sz[x] += sz[to];
    int pos = min(tot - sz[to], sz[to]);
    if(mx < pos) {
      bu = x, bv = to, bid = i;
      mx = pos;
    }
  }
}

void gotpoint(int x, int fa, P p) {
  if(x <= n) {
    pa[++cnt] = p;
```

```
  }
  for(int i = T.head[x]; i; i = T.Next[i]) {
    int to = T.to[i];
    if(us[i >> 1] || to == fa) {
      continue;
    }
    gotpoint(to, x, p + T.w[i]);
  }
}

void MakeConvex(int &num, P *q) {
  if(!cnt) {
    num = 0;
    return ;
  }
  sort(pa + 1, pa + 1 + cnt);
  q[num = 1] = pa[1];
  for(int i = 2; i <= cnt; ++i) {
    for( ; num >= 2; --num) {
      if((pa[i] - q[num - 1]) * (q[num] - q[num - 1])
> 0) {
        break;
      }
    }
    q[++num] = pa[i];
  }
}

void merge() {
  if(!na || !nb) {
    int L = max(na, nb);
```

```
    for(int i = 1; i <= L; ++i) {
      ans[++len] = na ? qa[i] : qb[i];
    }
    return ;
  }
  ans[++len] = qa[1] + qb[1];
  for(int u = 2, v = 2; u <= na || v <= nb; ) {
    if(u > na) {
      ans[++len] = qa[u - 1] + qb[v++];
    } else if(v > nb) {
      ans[++len] = qa[u++] + qb[v - 1];
    } else {
      P du = qa[u] - qa[u - 1], dv = qb[v] - qb[v -
1];
      if(du * dv < 0) {
        ans[++len] = qa[u++] + qb[v - 1];
      } else {
        ans[++len] = qa[u - 1] + qb[v++];
      }
    }
  }
}

void go(int x, int size) {
  if(size == 1) {
    return ;
  }
  tot = size, mx = bu = bv = 0;
  gotsz(x, 0);
  int x1 = bv, s1 = sz[bv], x2 = bu, s2 = tot - s1;
  us[bid >> 1] = 1;
```

```
  cnt = 0, gotpoint(x1, x2, P(0,0)), MakeConvex(na,
qa);
  cnt = 0, gotpoint(x2, x1, T.w[bid]), MakeConvex(nb,
qb);
  merge();
  go(x1, s1);
  go(x2, s2);
}

ll js(P x, ll y) {
  return x.x * y + x.y;
}

int main() {
  scanf("%d%d", &n, &m);
  now = n;
  for(int i = 1, x, y, A, B; i < n; ++i) {
    scanf("%d%d%d%d", &x, &y, &A, &B);
    S.adde(x, y, P(A, B));
  }
  for(int i = 1; i <= n; ++i) {
    fr[i] = i;
  }
  build(1, 0);
  go(1, now);
  cnt = len;
  memcpy(pa, ans, sizeof pa);
  MakeConvex(len, ans);
  for(int i = 0, cur = 1; i < m; ++i) {
    for( ; cur < len; ++cur)
      if(js(ans[cur], i) > js(ans[cur + 1], i)) {
```

```
            break;
        }
      printf("%lld", js(ans[cur], i));
      putchar(i < m - 1 ? ' ' : '\n');
    }
    return 0;
}
```