/\*

PROG: cowemb

LANG: C++

ID: hayk.sa1

\*/

#include

#include

#include

using namespace std;

#define eps (1e-9)

const double pi = 2\*acos(0.0);

int m = 1;

long long r;

struct interval

{

int i;

bool open;

double phi;

} d[100010];

int st[50010];

bool operator < (interval a, interval b)

{

return a.phi+eps < b.phi;

}

double get\_phi(double x, double y)

{

double ret;

if (x > eps)

ret = atan(y/x);

if (x >= -eps && x <= eps)

ret = ((y>0)\*2-1)\*pi/2;

if (x < -eps)

ret = pi-atan(-y/x);

if (ret < eps)

ret += 2\*pi;

return ret;

}

void compute(int i, long long a, long long b, long long c)

{

long long D;

double sd, co;

if (b)

{

D = a\*a\*c\*c-(a\*a+b\*b)\*(c\*c-r\*r\*b\*b);

if (D < 0)

return;

sd = sqrt(D+0.0);

co = (-a\*c+sd)/(a\*a+b\*b);

d[m++].phi = get\_phi(co, -(a\*co+c)/b);

co = (-a\*c-sd)/(a\*a+b\*b);

d[m++].phi = get\_phi(co, -(a\*co+c)/b);

if (d[m-1].phi > d[m-2].phi)

d[m-1].open = 0;

else

d[m-1].open = 1;

d[m-2].open = !d[m-1].open;

d[m-1].i = d[m-2].i = i;

}

else

{

D = b\*b\*c\*c-(a\*a+b\*b)\*(c\*c-r\*r\*a\*a);

if (D < 0)

return;

sd = sqrt(D+0.0);

co = (-b\*c+sd)/(a\*a+b\*b);

d[m++].phi = get\_phi(-(b\*co+c)/a, co);

co = (-b\*c-sd)/(a\*a+b\*b);

d[m++].phi = get\_phi(-(b\*co+c)/a, co);

if (d[m-1].phi > d[m-2].phi)

d[m-1].open = 0;

else

d[m-1].open = 1;

d[m-2].open = !d[m-1].open;

d[m-1].i = d[m-2].i = i;

}

}

int a[100010];

void add(int p, int v)

{

while (p <= m)

{

a[p] += v;

p += ((p^(p-1))+1)>>1;

}

}

int sum(int p, int q)

{

int ret = 0;

p--;

while (p > 0)

{

ret -= a[p];

p -= ((p^(p-1))+1)>>1;

}

while (q > 0)

{

ret += a[q];

q -= ((q^(q-1))+1)>>1;

}

return ret;

}

int main()

{

freopen("cowemb.in", "r", stdin);

freopen("cowemb.out", "w", stdout);

long long ans = 0;

int n, a, b, c, i;

scanf("%d%lld", &n, &r);

for (i = 0; i < n; i++)

{

scanf("%d%d%d", &a, &b, &c);

compute(i, a, b, c);

}

sort(d+1, d+m);

for (i = 1; i <= m-1; i++)

if (d[i].open)

{

add(i, 1);

st[d[i].i] = i;

}

else

{

ans += sum(st[d[i].i]+1, i);

add(st[d[i].i], -1);

}

printf("%lld\n", ans);

return 0;

}