#include <cstdio>

#include <algorithm>

using namespace std;

#define MAX 100010

#define MAXM 75010

int n, m, d;

///////////////////// 1-dimenison

int p1[MAX];

long long solve1()

{

int s, t, i;

long long ans=0;

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

scanf("%d", p1+i);

sort(p1, p1+n);

for (s = t = 0; s+1 < n; s++)

{

while (t < n && p1[t]-p1[s] <= d)

t++;

t--;

ans += t-s;

}

return ans;

}

///////////////////// 2,3-dimension

int T[2\*MAXM];

void build(int b)

{

int i;

for (i = 1; i <= m\*b; i++)

T[i] = ((i^(i-1))>>1)+1;

}

///////////////////// 2-dimenison

#define MAX2 75010

struct twoD

{

int p, q;

} p2[MAX];

bool operator < (twoD a, twoD b)

{

return a.p < b.p;

}

int s2[2\*MAX2];

void add2(int p, int a)

{

p += m;

while (p <= (m<<1))

{

s2[p] += a;

p += T[p];

}

}

int sum2(int p)

{

int ret=0;

p += m;

if (p > (m<<1))

p = (m<<1);

while (p > 0)

{

ret += s2[p];

p -= T[p];

if (T[p] == 0 && p)

p = p;

}

return ret;

}

long long solve2()

{

long long ans=0;

int x, y, s, t, i;

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

{

scanf("%d%d", &x, &y);

p2[i].p = x+y;

p2[i].q = x-y;

}

sort(p2, p2+n);

build(2);

for (s = t = 0; t < n; t++)

{

while (p2[t].p-p2[s].p > d)

add2(p2[s++].q, -1);

ans += sum2(p2[t].q+d)-sum2(p2[t].q-d-1);

add2(p2[t].q, 1);

}

return ans;

}

///////////////////// 3-dimenison

#define MAX3 80

#define Min(a, b) ((a)<(b)?(a):(b))

struct threeD

{

int p, q, w, t;

} p3[MAX];

bool operator < (threeD a, threeD b)

{

return a.t < b.t;

}

int s3[3\*MAX3][3\*MAX3][3\*MAX3];

void add3(int p, int q, int w, int a)

{

int i, j, k;

p += m;

q += m;

w += 2\*m;

for (i = p; i <= m\*3; i+=T[i])

for (j = q; j <= m\*3; j+=T[j])

for (k = w; k <= m\*3; k+=T[k])

s3[i][j][k] += a;

}

int sum3(int p, int q, int w)

{

int ret=0, i, j, k;

for (i = p; i > 0; i-=T[i])

for (j = q; j > 0; j-=T[j])

for (k = w; k > 0; k-=T[k])

ret += s3[i][j][k];

return ret;

}

long long solve3()

{

long long ans=0;

int x1, y1, z1, x2, y2, z2, s, t, i;

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

{

scanf("%d%d%d", &x1, &y1, &z1);

p3[i].t = x1+y1+z1;

p3[i].p = x1+y1-z1;

p3[i].q = x1-y1+z1;

p3[i].w = x1-y1-z1;

}

sort(p3, p3+n);

build(3);

for (s = t = 0; t < n; t++)

{

while (p3[t].t-p3[s].t > d)

{

add3(p3[s].p, p3[s].q, p3[s].w, -1);

s++;

}

x2 = Min(p3[t].p+d, 2\*m)+m;

y2 = Min(p3[t].q+d, 2\*m)+m;

z2 = Min(p3[t].w+d, m)+2\*m;

x1 = Min(p3[t].p-d-1, 2\*m)+m;

y1 = Min(p3[t].q-d-1, 2\*m)+m;

z1 = Min(p3[t].w-d-1, m)+2\*m;

ans += sum3(x2,y2,z2)-sum3(x2,y2,z1)-sum3(x2,y1,z2)-sum3(x1,y2,z2)+

sum3(x2,y1,z1)+sum3(x1,y2,z1)+sum3(x1,y1,z2)-sum3(x1,y1,z1);

add3(p3[t].p, p3[t].q, p3[t].w, 1);

}

return ans;

}

int main()

{

int b;

scanf("%d%d%d%d", &b, &n, &d, &m);

if (b == 1)

printf("%lld\n", solve1());

if (b == 2)

printf("%lld\n", solve2());

if (b == 3)

printf("%lld\n", solve3());

return 0;

}