Nonlinear system

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	512 megabytes

Congratulations! You are chosen to build software for a new GPS/GLONASS receiver.

The satellite navigation works as follows (of course, this is a rather simplified description of a real-world situation). There are $N \leq 30$ satellites. Every satellite broadcasts its position (x_i, y_i, z_i) and high-precision synchronized time t_i. These signals take time to reach the receiver (*e.g.*, the one that's in your smartphone). If the receiver has position x, y, z, and receives the signal at time t, the following equation (called "Navigation equation") holds true[1]:

$$(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 = (t - t_i)^2$$

As we can see, we have four unknowns (the position of the receiver x, y, z and the precise time t when it received the signal). So, we need at least $N = 4$ satellites to get the location of the receiver ("fix").

The system of exactly four navigational equations might, in general, have multiple solutions[2]. But usually, more than $N > 4$ satellites are visible, and we have an overdetermined system of non-linear equations (due to noise, the equations can not be satisfied exactly). In this case, our goal is to minimize the sum of squares of residuals: $\sum r_i^2(x, y, z, t) \to \min$, where

$$r_i(x, y, z, t) = (x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 - (t - t_i)^2.$$

Your program should continuously read data from a virtual GPS receiver and print the position in each moment until the signal is lost. The number of satellites (and their order) can change. The initial position is unknown, but the position between successive readings does not change too much. The required accuracy is given by $\sum r_i^2(x, y, z, t) < 10^{-6}$. It is guaranteed that such solution exists.

The number of satellites $N \leq 30$, the coordinates x, y, z are in range $[-10; 10]$, the time t is in range $[-1000; 1000]$. The number of readings is guaranteed not to exceed 10^5.

As usual, you have several blackbox functions:

- `void gps_init()` – initializes the internal blackbox data structures. Should be called at the very beginning of the program! No other blackbox function should be called before it, and no reading from the `stdin` shall occur.
- `int gps_read(double *x, double *y, double *z, double *t)` – reads data from the receiver. The data is written to the arrays `x`, `y`, `z`, `t`, and the function returns the number of satellites for which we have the data. The arrays `x`, `y`, `z`, `t` must be allocated *by you*, and contain at least 30 elements each. For example, if the function returns 5, that means that first five elements of `x` now contain x coordinates of the five satellites, *etc.* If `gps_read` returns any value less than four, it means that we have lost the signal, and the program should quit (`return 0;`).
- `void gps_submit(double x, double y, double z, double t)` – accepts calculated coordinates of the receiver. Must be called after every successful (return value ≥ 4) call to `gps_read`.

[1] For simplicity, we choose units of distance and time such that the speed of light $c = 1$.

[2] `https://doi.org/10.1109/7.104271`