OTTO Application Description

The requirements for OTTO Application are as below.
A course is set up in a 100m by 100m factory. Certain points are identified within the space as waypoints
to visit to receive goods. They are ordered -- there is a waypoint 1, a waypoint 2, etc. Your robot must start at (0,0). From there, it should go to waypoint 1, stop for 10 second, go to waypoint 2, stop for 10 second, and so on. It must finally end up at, and stop for 10 second on
(100,100).

Each waypoint except (0,0) and (100,100) has a time penalty for missing the load to reflect the time needed for a human to handle the work later.
So, if your OTTO went straight from waypoint 1 to waypoint 3, skipping waypoint 2, it would incur waypoint 2's penalty. Note that once it hits waypoint 3, it cannot go back to waypoint 2. It must hit the waypoint in order. Since your OTTO must stop for 10 seconds on each waypoint to be loaded, it is not in danger of hitting a waypoint accidentally too soon. For example, if waypoint 3 lies directly between waypoints 1 and 2, your OTTO can go straight from 1 to 2, right over 3, without stopping.
Since it didn't stop to be loaded, waypoint 3's penalty will not be incurred.
Your final score is the amount of time (in seconds) your OTTO takes to reach (100,100), completing the course, plus all penalties. Smaller scores are better.
OTTO is very manoeuvrable, but a bit slow. It moves at 2 m/s, but can turn very quickly. During the 10 seconds it stops on a waypoint point, it can easily turn to face the next waypoint. Thus, it can always move in a straight line between target points.
Because OTTO is a bit slow, it might be advantageous to skip some waypoints and incur their penalty, rather than actually manoeuvring to them.
Given a description of a course, determine OTTO's best (lowest) possible time.

Input to OTTO application

There will be several test cases fed into this OTTO application-solution via stdin. Each test case will begin with a line with one integer, N (1 <= N <= 1000) which is the number of waypoints on the course.
Each of the next N lines will describe a waypoint with three integers, X, Y and P, where (X, Y) is a location on the course (1 <= X, Y <= 99, X and Y in meters) and P is the penalty incurred if OTTO misses that waypoint (1 <= P <= 100).
The waypoints will be given in order -- the first line after N is target 1, the next is waypoint 2, and so on. All the waypoints on a given course will be unique -- there will be at most one waypoint point at any location on the course.
End of input will be marked by a line with a single `0'.

Output from OTTO application

For each test case, output a single decimal number, indicating the smallest possible time for that course. Output this number rounded (NOT truncated) to three decimal places. Print each answer on its own line on stdout and do not print any blank lines between answers

Sample Input

```
1
50 50 20
3
30 30 90
60 60 80
10 90 100
3
30 30 90
60 60 80
10 90 10
0
```

Sample Output

```
90.711
156.858
110.711
```