The Playground (Implementation - Competition)

The Student Entertainment Committee at DTU has just been granted a huge amount of money dedicated for building a new playground for DTU students. The committee has already decided the kind of playground, but they need your help to actually design the playground.

The playground consists of a number of *platforms*. Two of the platforms are special; the *start-platform* is the only place where students can enter the playground and the *end-platform* is the only platform where students can leave. Between a pair of platforms there can be different kinds of *activities* such that when you do the activity you will move from the first platform to ther other one. The possible activities are listed below including their restrictions:

The NEWS[1] Slide (A1). Can be built from a platform A to any other platform on the same vertical or horizontal line as A.
The Human Cannon (A2). Can be built from a platform A to the platform furthest away from A (euclidean distance). If multiple platforms are equally far away, it can only be built to the *first* of these (ie. the first platform in the order they come in the input)
The Platform Trampoline (A3). Can be built from a platform A to any other platform B if there are at least two other platforms in between A and B (ie. exactly on the line segment between A and B there must be at least 2 other platforms).
The EOF[2] Wormhole (A4). Can be built from any platform directly to the end-platform.

The playground is divided into different *trails*. A trail is a sequence of activities that starts at the start-platform and ends at the end-platform. Two trails cannot have any activities in common, but they may share platforms.

Based on the terrain, it has already been decided where the platforms should be located. Your task is to decide which activities should be built between each pair of platform. Multiple activities may be built between the same two platforms - even activities of the same kind. There is a limit on how many of each kind of activity can at most be built from each of the platforms. Your target is to design the playground with as many trails as possible.

Input The input has the following format. All numbers are integers:

```
Line 1: "P". P is the number of platforms (2 <= P <= 100.000).
Next P lines: "X Y A1 A2 A3 A4". (X, Y) is the position of the platform.
              A1, A2, A3, and A4 are the restrictions on the number of
              each kind of activity starting from the platform.
              (0 <= X, Y <= 100.000 and 0 <= A1, A2, A3, A4 <= 100.000)
```

The first platform in the input is the start-platform and the last platform is the end-platform. You can assume no two platforms have the same position. Your program should read the input from standard input.

Output Your program should output the maximum possible number of trails the playground can have (to standard output).

Sample test This is an example of the input. The answer for this test is 5 (see Figure 1):

[1] North, East, West, South
[2] End-of-Fun

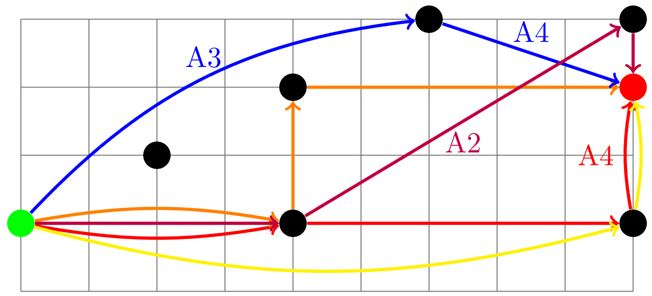

Figure 1: Illustration of the first sample test with an optimal solution. Platforms are marked by circles. The green circle is the start-platform. The red circle is the end-platform. An arrow corresponds to an activity between two platforms. The label on an arrow tells the type of activity in this solution (arrows without labels are all A1). This solution has 5 trails (colored in different colors). This solution is optimal, but other optimal solutions may exist.

```
8
0 1 4 0 2 0
2 2 1 2 3 4
4 1 3 1 1 0
4 3 1 0 0 1
6 4 0 0 1 1
9 1 1 0 0 2
9 4 2 0 0 1
9 3 0 0 0 0
```

More sample tests can be found on CodeJudge.

Technical details You can assume your program can use at least 200MB of heap memory and 100MB of stack memory. We may likely choose to only use smaller test cases than the constraints above suggest, but your program should support it, if some solutions are so fast we cannot determine the winners otherwise (so don't be too concerned if your program is slow on such large inputs). We will both use randomly generated test data and test data that tests for special situations. You can use any of the languages C/C++/Java/C#/Python/F#.

Hand-in You are allowed to solve the exercise in teams of 1-3 students. You must hand-in your solution before **November 25, 23.59** on CodeJudge (`https://dtu.codejudge.net/02110-e15/assignment/show/1204`). Hand-in under the exercise "The Playground (mandatory)". If you solve the exercise (get a green smiley on CodeJudge), *it will count as a passed mandatory assignment for each of the team members.*

Competition You can submit your solution to a competition among the students in the course. The rank in the competition will be determined based on (1) the number of solved test cases (2) the total sum of your programs running time on all test cases it solved. (2) will only apply, if two teams solve the same number of test cases. The competition will be based on different test data than the tests we provide before the deadline. To join the competition, please **also** hand-in your solution on CodeJudge under the exercise "Competition".