DAKI Programmeeropdracht 3: Air Traffic Control

(Deze versie is van 2 mei 2020, 18:25)

Je moet deze programmeeropdracht **zelf** en **alleen** maken. Je mag zeker met anderen overleggen over je aanpak, maar code van anderen bekijken of overnemen of zelf code delen met anderen is uitdrukkelijk niet toegestaan. Je moet je programma schrijven in C#, en inleveren via DOMjudge.

1 Opdrachtbeschrijving

DAKIBOT vindt het maar niks, werken in een vieze fabriek, met lijm en zo... gatver! Ze wil liever een wat... 'intellectueel uitdagender' bijdrage aan de maatschappij leveren. Ze is zich ervan bewust dat veel onderzoek waaruit haar voorouders zijn voortgekomen werd gefinancierd door subsidies van overheidsinstanties, met name defensie instituten, zoals DARPA met hun Robotics Challenge, maar ook private ondernemingen zoals Boston Dynamics. Iets met vliegen lijkt haar wel wat, maar dan toch liever de burgerluchtvaart dan militaire toestanden. Opeens heeft ze het! Ze wordt luchtverkeersleider!

De taak van een luchtverkeersleider is om al het luchtverkeer veilig door het luchtruim te navigeren, door piloten instructies te geven omtrent hun te volgen koers. Een belangrijk doel is dat vliegtuigen niet te dicht bij elkaar in de buurt komen. DAKIBOT heeft dus een correct en *snel* algoritme nodig, dat voor een gegeven verzameling (x, y)-coördinaten van vliegtuigen in een bepaald rechthoekig gebied, bepaalt wat de kleinste onderlinge afstand is die er tussen twee van die vliegtuigen bestaat. Ze kan dan een menselijke operator ondersteunen door hem of haar te wijzen op zo'n paar, waar het risico het grootst is, en een gesprekje met de piloten geboden is.

2 Invoer en Uitvoer

De eerste regel van de input bevat een geheel getal $2 \leq n \leq 1\,000\,000$, het aantal vliegtuigen op het scherm. Daarna volgen n regels met op elke regel de positie van een vliegtuig, in de vorm van een x- en een y-coördinaat. Dit zijn niet-negatieve gehele getallen $\leq 10\,000\,000$, gescheiden door een spatie. Als output geef je *het kwadraat van* de minimale afstand tussen twee posities in de verzameling van n posities. Je moet dit kwadraat toch berekenen als je de afstand wil weten, maar in plaats van de wortel te trekken en afrondingsfouten te introduceren, vragen we enkel om het kwadraat van de afstand.

3 Voorbeeld

Bij deze invoer:

```
10      // 10 vliegtuigen
34 22   /// eerste vliegtuig
73 46   /2de vliegtuig
30 50   % et
31 24   $ cetera
34 63
28 79
24 26
10 81
42 52
95 5   dat was het alweer!
```

hoort de volgende uitvoer:

```
13
```

4 Algoritmische eisen

Je *moet* recursie gebruiken om een algoritme te ontwerpen dat snel genoeg is. In de werkcollegeopdrachten over recursie staat een vraag om je hierbij te helpen.

5 Hints, tips

Geen.