Mooie priem

Euclides en Mersenne hebben een discussie over priemgetallen en het verband met machten van 2. Helaas geraken ze er niet uit, en moet jij hen helpen. Merk op, het getal 1 is géén priemgetal, gezien we de volgende definitie hanteren: "Een natuurlijk getal is een priemgetal als en slechts als het exact twee verschillende natuurlijke getallen als delers heeft, namelijk 1 en zichzelf.".

Opgave

Genereer het kleinste priemgetal p tussen de grenzen m en n ($m \leq p \leq n$) dat het dichtst bij een macht van 2 ligt. Indien er geen priemgetal in het bereik ligt, moet de boodschap "`geen priemgetal gevonden`" uitgeprint worden. Merk op dat m en n strikt positieve natuurlijke getallen zijn die maximaal $1,000,000$ bedragen.

Invoer

De eerste lijn van de invoer bevat het aantal testgevallen N, dus het aantal intervallen dat we gaan onderzoeken. Daarna volgens dus N lijnen, met op elke lijn 2 waarden gescheiden door één spatie: eerst m en daarna n waarbij $m \leq p \leq n$.

Uitvoer

Het programma dient de priemgetallen uit te schrijven. Elk priemgetal staat op een aparte lijn, en voor elke lijn in de invoer is er een lijn in de uitvoer (behalve voor de eerste lijn, die het aantal testcases bevat).

Voorbeeld

Invoer

```
3
14 20
24 26
40 100
```

Uitvoer

```
17
geen priemgetal gevonden
61
```

Bonus

Als een efficiënt programma voor deze opgave gevonden wordt, dan kan een bonus van 60 minuten worden verdiend die afgetrokken wordt van de totale submissietijd.

Een programma wordt als efficiënt genoeg beschouwd als het in staat is om snel het kleinste priemgetal dat het dichtst bij een macht van 2 ligt te vinden, ook als de grenzen m en n veel groter zijn dan $1,000,000$.

Merk op dat deze opgave eerst correct moet opgelost worden voor kleine waarden van m en n, voordat er kans kan gemaakt worden op de bonus. Foutieve pogingen om de bonus te verdienen leveren *geen* extra straftijd op.

Tegels

Piet Precies is een vloerder die altijd werkt met volledige tegels, en die dus nooit tegels versnijdt om zijn vloer af te werken. Piet werkt enkel met klanten die een rechthoekige vloer willen betegeld zien, en geen bezwaar hebben dat de tegels geen vast patroon vormen. De klant kiest de tegels (materiaal, vorm, ...), die typisch enkel in een beperkt aantal vormen beschikbaar zijn. Van elke vorm is een onbeperkte voorraad tegels beschikbaar.

Opgave

Schrijf een programma dat, aan de hand van de afmetingen van de rechthoekige vloer en van de beschikbare tegelvormen aangeeft of meneer Precies de vloer volledig kan betegelen zonder tegels te moeten versnijden. Overlappende tegels zijn daarbij geen optie, het draaien van tegels over 90 graden is uiteraard wel toegestaan. De afmetingen van de vloer en van de tegels worden uitgedrukt in cm, waarbij de lengte en de breedte van zowel de vloer als de tegels strikt positieve natuurlijke getallen zijn.

Indien de vloer volledig kan betegeld worden met tegels van de beschikbare tegelvormen (zonder tegels te versnijden of tegels te laten overlappen), moet de boodschap "`JA`" uitgeprint worden. In het andere geval moet "`NEEN`" worden uitgeprint.

Invoer

De invoer bestaat uit een aantal bestellingen. Voor elke bestelling worden de afmetingen gegeven van de vloer, en van de beschikbare tegelvormen. De eerste lijn van de invoer bevat één geheel getal n, dat aangeeft voor hoeveel bestellingen een oplossing moet bepaald worden.

Per bestelling wordt de nodige informatie weergegeven op verschillende lijnen. Eerst worden de afmetingen van de vloer aangegeven door twee getallen, gescheiden door één spatie. De daaropvolgende lijn bevat één geheel getal k, het aantal beschikbare tegelvormen. Daarop volgen k lijnen, die telkens de afmetingen van de tegels aanduiden, opnieuw door twee getallen, gescheiden door één spatie. De informatie voor de verschillende bestellingen volgt direct op elkaar, zonder enige scheidingstekens of -lijnen.

Uitvoer

De uitvoer bestaat uit n lijnen, die elk de boodschap "`JA`" of "`NEEN`" bevatten.

Voorbeeld

Invoer

```
4
50 70
3
20 30
30 30
20 40
50 50
2
30 30
20 20
50 50
3
30 30
20 20
30 20
100 20
1
30 20
```

Uitvoer

```
JA
NEEN
JA
NEEN
```

Som van priemgetallen

Priemgetallen zijn natuurlijke getallen die voldoen aan de alom gekende definitie:

"Een natuurlijk getal is een priemgetal als en slechts als het exact twee verschillende natuurlijke getallen als delers heeft, namelijk 1 en zichzelf."

Merk op: het getal 1 is dus *geen* priemgetal!

Priemgetallen zijn niet enkel wiskundig interessant om verschillende redenen, maar hebben ook veel bruikbare toepassingen, waaronder het versleutelen van geheime berichten.

Een wiskundige eigenschap, die eenvoudig te bewijzen is, zegt dat elk getal groter dan 1 kan geschreven worden als een som van (een of meer) priemgetallen.

Opgave

Schrijf een programma dat voor een gegeven natuurlijk getal x het ***kleinste aantal*** priemgetallen kleiner dan 100 zoekt waarvan de som gelijk is aan x. Indien er meerdere combinaties van priemgetallen zijn die aan deze voorwaarde voldoen, dan krijgt de combinatie met het grootste niet-gemeenschappelijke priemgetal voorrang op de andere. Zo b.v. zal voor $x = 20$ de oplossing $17 + 3$ voorrang krijgen op $13 + 7$.

Invoer

De eerste regel van de invoer bestaat uit een getal n dat aangeeft hoeveel getallen volgen. De n daaropvolgende regels bevatten telkens één getal, kleiner dan 1000, waarvoor de oplossing moet bepaald worden.

Uitvoer

De uitvoer bestaat uit n lijnen, één per getal x waarvoor de som van priemgetallen moet bepaald worden. Elke oplossing wordt uitgeprint in de vorm:

$$p_1 + p_2 + \ldots + p_k$$

waarbij $p_1 \geq p_2 \geq \ldots \geq p_k$. Er worden *geen spaties* gebruikt tussen de getallen en de optellingstekens.

Voorbeeld

Invoer

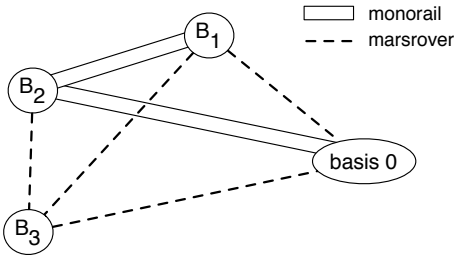
```
5
20
123
37
106
293
```

Uitvoer

```
17+3
97+23+3
37
89+17
97+97+97+2
```

Optimale Marsverbindingen

Op Mars zijn een aantal marsbasissen gebouwd waarvan de liggingen gekozen zijn op basis van onderzoeks- of mijningsmogelijkheden. Basis 0 is de hoofdbasis. Hier komen alle vluchten van Aarde aan. Op dit ogenblik verloopt al het vervoer tussen basis 0 en een andere basis met marsrovers (in een rechte lijn), een soort rupsvoertuigen die 20 km/u rijden. Er wordt overwogen om een aantal buizen met een monorail te bouwen tussen enkele basissen. Daarop kan dan een trein rijden aan 200 km/u. Aangezien er maar een beperkt budget is kan er maar een beperkt aantal ($= k$) km rail worden aangelegd.



Figuur 1: Voorbeeld van een aantal marsbassisen.

Opgave

Schrijf een programma dat bepaalt welke verbindingen tussen 2 basissen vervangen kunnen worden door een rail zodat de totale tijd om vanuit basis 0 elke andere basis te bezoeken minimaal is. De tijd om een andere basis te bezoeken wordt steeds gerekend vertrekkende vanuit basis 0, waarbij eventueel andere basissen kunnen gepasseerd worden. Voor het berekenen van afstanden wordt de Euclidische afstand gebruikt ($d(x, y) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$) en wordt er gerekend met floating-point getallen. Ook voor tijden wordt met floating-point getallen gewerkt. Het is dus de bedoeling om maximum k km rail te leggen tussen basissen zodat

$$\sum_{i=1}^{n} tijd(B_0, B_i)$$

minimaal is. Het resultaat, zijnde de minimale totale tijd, moet naar een geheel getal worden omgevormd (zie uitvoer).

Invoer

De eerste lijn van de invoer duidt het aantal testgevallen T aan. Per testgeval volgt dan een lijn met twee natuurlijke getallen, gescheiden door één spatie. Het eerste getal n duidt het aantal marsbasissen aan ($2 \leq n \leq 10$, inclusief marsbasis 0), het tweede getal k komt overeen met het totaal aantal km rail waarvoor er budget kan vrijgemaakt worden ($0 \leq k \leq 100$). Daarop volgen n lijnen met de x- en y-coördinaten van de marsbasissen (steeds gescheiden door één spatie). De coördinaten zijn steeds natuurlijke getallen ($0 \leq x_i \leq 100$ en $0 \leq y_i \leq 100$) en liggen steeds exact op kilometerpunten. Met andere woorden, de basis op coördinaten (2,1) ligt exact op 1 kilometer afstand van de basis op coördinaten (2,2).

Je hoeft geen rekening te houden met kruisende verbindingen, noch van marsrovers, noch van monorails of monorail en marsrovers met elkaar.

Uitvoer

Als uitvoer dient de minimale totale tijd in minuten per testgeval uitgeprint worden. Hierbij wordt de totale tijd naar boven afgerond naar een geheel aantal minuten.

Voorbeeld

Invoer

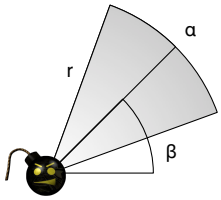
```
2
3 1
0 0
0 1
0 2
4 5
0 0
0 5
1 3
1 6
```

Uitvoer

```
4
16
```

Opstand in Flatland

Flatland. Ooit was het een vredige tweedimensionale wereld. Alle vlakke figuren van diverse strekkingen leefden gelukkig samen. De laatste jaren is er echter veel veranderd. De Cirkels grepen de macht en installeerden een totalitair regime, onder het mom van 'veiligheid'. Regelmatige veelhoeken bevolkten de sociale ladder en het leger. Alle andere figuren waren van weinig of geen waarde meer. De intellectuele Concaven werden genadeloos vervolgd, met als gevolg dat ze onderdoken en een verzetsbeweging startten. Met vrij rudimentaire middelen bouwden ze toch een intelligente bom, zie Figuur 2.



Figuur 2: Een slimme bom en zijn kenmerken: (i) de hoek van de actieradius $0 < \alpha < 360$, (ii) de hoek $0 < \beta < 360$ tussen de bissectrice van de actieradius en de horizontale X-as, en (iii) de straal r van de actieradius.

De explosie veroorzaakt door deze bom neemt de vorm aan van een tweedimensionaal taartstuk, met een zekere actieradius $r > 0$ en een hoek $0 < \alpha < 360$ (alle hoeken worden in graden uitgedrukt). Als de bom ontploft, wordt alles in zijn bereik onmiddellijk verpulverd. Dit betekent dat de ontploffing door geen enkel object kan worden tegengehouden; het heeft dus geen nut dekking te zoeken achter een medesoldaat. Een soldaat wordt geneutraliseerd indien een van de volgende voorwaarden voldaan is.

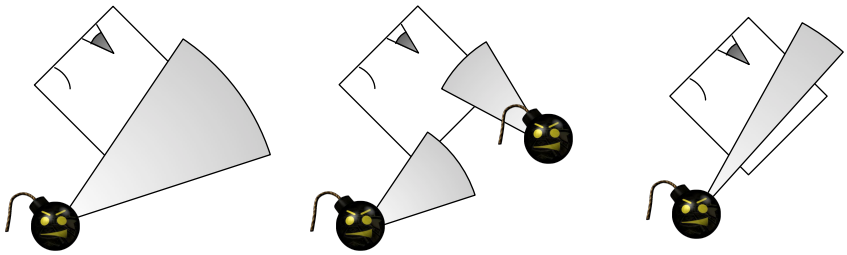
1. Minstens één zijde is volledig vernietigd, dat wil zeggen als twee (of meer) adjacente hoekpunten beiden binnen een bomradius liggen.

2. De bom klieft de soldaat in twee delen. Hiermee bedoelen we dat één bom twee of meer zijden van de soldaat snijdt, en wel zo dat de soldaat in twee niet samenhangende delen gesplitst wordt.

Om de opgave niet nodeloos te compliceren worden de drie mogelijkheden waarmee je rekening dient te houden voorgesteld in Figuren 3 (a)-(c). Voor alle duidelijkheid, je hoeft een soldaat enkel als doorkliefd te beschouwen als hij door één enkele bom in twee delen wordt gesplitst. Je hoeft dus

geen rekening te houden met eventuele overlappende bommen, tenzij deze natuurlijk elk een of meerdere hoeken afhakken van de soldaat.

.



(a) Zijde af door 1 bom (b) Zijde af door 2 bommen (c) Doorkliefd door 1 bom

Figuur 3: De drie manieren waarop een soldaat kan geneutraliseerd worden: (a) een bom hakt een zijde af, (b) verschillende bommen hakken verschillende (adjacante) hoeken af, (c) de actieradius snijdt de veelhoek met beide explosiezijden.

Opgave

Schrijf een programma dat bepaalt welke soldaten geneutraliseerd worden voor een gegeven opstelling. De configuratie beschrijft de lokatie van de bommen (x, y), hun actieradius r, actiehoek α en orientatie bepaald door β. Alle hoeken worden uitgedrukt in graden. De orientatie van een bom is de hoek tussen de bissectrice van het taartstuk en de horizontale X-as. De configuratie beschrijft ook de soldaten, waarbij hun zwaartepunt (x, y) (het centrum van de omschrijvende cirkel), de diameter d van de omschrijvende cirkel en hun aantal hoeken h worden gegeven. Je mag ervan uitgaan dat elke soldaat een hoekpunt op de coördinaat $(x + d/2, y)$ heeft. Alle bommen ontploffen tegelijkertijd.

Invoer

De invoer bestaat uit een aantal regels die via standaard invoer gelezen worden. Op de eerste regel staat het aantal configuraties die we beschouwen. Elke configuratie bestaat uit een aantal regels, met op de eerste regel twee natuurlijke getallen: het aantal bommen b en het aantal soldaten s. Vanaf de tweede regel van de configuratie volgen b lijnen met op elke lijn de beschrijving van een bom. Een dergelijke lijn bestaat uit 5 natuurlijke getallen: x y α β r. Daarna volgen er s lijnen met natuurlijke getallen die elke een soldaat beschrijven: x y d h. Onmiddellijk daarna volgt de eerste

regel van de volgende configuratie, die dus opnieuw uit twee getallen bestaat. Alle getallen worden door exact één spatie gescheiden indien ze op dezelfde regel voorkomen.

Uitvoer

Het programma geeft voor elke soldaat terug of hij geneutraliseerd wordt, door "`dood`" of "`levend`" op een lijn te printen. Hierbij is de volgorde natuurlijk van belang. Er dient *geen* regel te worden gevoegd tussen twee configuraties, dus alle soldaten van alle configuraties volgen elkaar onmiddellijk op in de uitvoer.

Voorbeeld

Invoer

```
3
1 2
-7 3 10 5 20
7 4 8 8
0 0 4 4
1 1
0 0 20 4 10
9 0 8 8
2 1
4 3 24 315 7
16 -1 20 180 9
9 0 8 8
```

Uitvoer

```
dood
levend
levend
dood
```