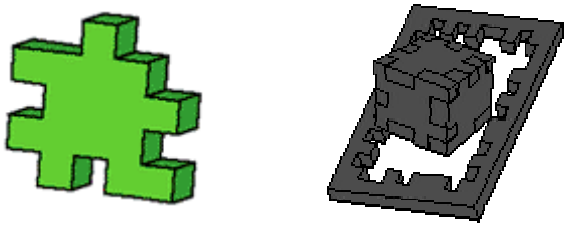
Beschrijving:

Je kent ze wel van vroeger; de wirrel warrel kubus. Een matje van kunststof waarin zes delen in allerlei vormen gesneden zijn. Ze zijn er in verschillende moeilijkheidsgraden. Van deze 6 delen kun je dan een kubus maken of je kunt andere delen samenvoegen tot één grote kubus. In figuur 1 is een afbeelding van een deel en een van een kubus weergegeven:

Figuur 1



Probleem:

Gegeven; zesmaal een matrix die een zijde van de kubus representeert (één van de zes delen uit het matje), dient er gecontroleerd te worden of er van de zes delen een sluitende kubus kan worden gemaakt. De kubus mag dus geen gaten bevatten.

Invoer: **Bestand: A.IN**

Een regel met het aantal runs. Vervolgens per run; voor 6 vlakken een regel met de lengte *n* (met $n \leq 100$) van een zijde van een vlak. Dan volgen voor ieder vlak *n* regels met ieder n cijfers, gescheiden door een spatie. Deze *n* regels representeren het vlak door middel van een 1 als het element gevuld is of een 0 als dat niet het geval is.

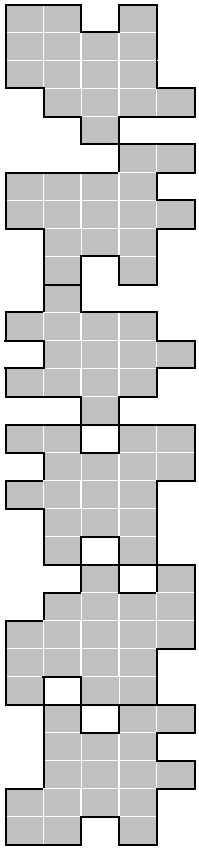
Uitvoer: **Bestand: A.OUT**

Voor iedere run een regel met daarop een ja als er van de gegeven vlakken een kubus gevormd kan worden of een nee indien dat niet het geval is.

Voorbeeld:

Bij de invoer: (Zie voorbeeld A.IN)

```
2
5
1 1 0 1 0
1 1 1 1 0
1 1 1 1 0
0 1 1 1 1
0 0 1 0 0
0 0 0 1 1
1 1 1 1 0
1 1 1 1 1
0 1 1 1 0
0 1 0 1 0
0 1 0 0 0
1 1 1 1 0
0 1 1 1 1
1 1 1 1 0
0 0 1 0 0
1 1 0 1 1
0 1 1 1 1
1 1 1 1 0
0 1 1 1 0
0 1 0 1 0
0 0 1 0 1
0 1 1 1 1
1 1 1 1 1
1 1 1 1 0
1 0 1 1 0
0 1 0 1 1
0 1 1 1 0
0 1 1 1 1
1 1 1 1 0
1 1 0 1 0
3
0 1 0
1 1 0
1 0 1
1 1 1
0 1 1
1 0 1
1 1 1
1 0 1
1 1 1
1 1 1
1 1 0
1 1 1
1 1 0
1 1 1
1 1 1
```



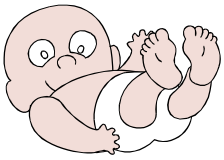
hoort de uitvoer: (A.OUT)

```
ja
nee
```

Babynamen — Opgave B

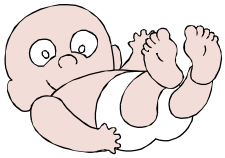
Beschrijving:

Hannah is al zeven maanden zwanger en heeft net een echoscopie laten maken waaruit bleek dat ze niet één maar twee wiegjes klaar moet zetten: er is namelijk een tweeling op komst.



In Hannah's familie is het gebruikelijk om je kind een palindroom-naam te geven, oftewel dat als je de naam van achter naar voren leest, dat je dan precies hetzelfde leest. Zo heet haar zus Ada, haar broer Bob en haar moeder Anna.

Maar omdat een tweeling wel heel speciaal is wil ze iets anders: als je de naam van de ene omgekeerd leest moet je de andere lezen en vice versa. 'Ot' & 'To' is hier een voorbeeld van.



Omdat Hannah wat moeite heeft met het vinden van namen die aan deze voorwaarde voldoen, roept ze hierbij jou hulp in.

Probleem:

Gegeven een lijst met n namen (n<=4000). Geef alle mogelijke combinaties van twee verschillende namen die elkaars omgekeerde zijn qua schrijfwijze.

Natuurlijk mag elke mogelijkheid maar één keer genoemd worden.

Invoer: **Bestand: B.IN**

Een regel met het aantal runs. Vervolgens per run een regel met n = het aantal namen. Dan n maal een regel met een naam van maximaal 15 letters, waarbij alleen kleine letters gebruikt worden.

Uitvoer: **Bestand: B.OUT**

Voor iedere run een regel met daarop het nummer van de run en dan per gevonden combinatie een regel met daarop beide namen gescheiden door een spatie.

Elke combinatie mag maar één keer voorkomen en zowel alle combinaties als de hele lijst moet alfabetisch gesorteerd worden..

Als er niets gevonden wordt, dan moet de regel 'niets gevonden' naar het bestand geschreven worden.

Voorbeeld:

Bij de invoer: (Zie voorbeeld B.IN)

```
3
4
ot
to
pa
ap
3
jan
piet
klaas
4
cba
abc
abc
cba
```

hoort de uitvoer: (B.OUT)

```
1
ap pa
ot to
2
niets gevonden
3
abc cba
```

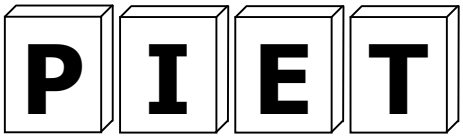
Beschrijving:

Vijf december is de dag van Sinterklaas
Dan gaat hij rond en geeft iedereen speculaas

Voor jou heeft de Sint een kado, o zo speciaal
Jouw naam in chocoladeletters, helemaal

Jij bent vast wel blij, maar Zwarte Piet
Die moest dat inpakken, dus was hij dat niet

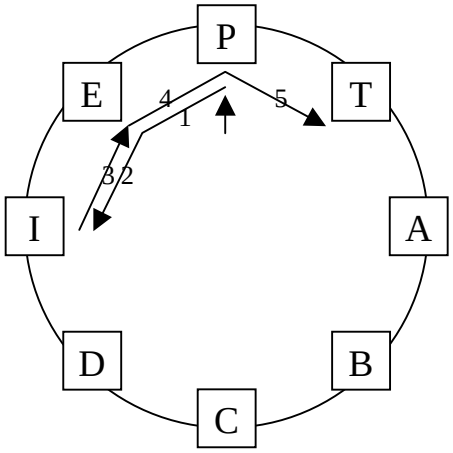
Alle letters stonden namelijk doorelkaar
En dan is het natuurlijk vrij zwaar.



Probleem:

In een grote loods staan in een grote cirkel allemaal dozen met chocoladeletters opgesteld. Elke doos bevat één soort (één letter) chocoladeletters, maar de dozen zijn willekeurig gerangschikt. Ook kunnen er meer dozen van één letter zijn.

Zwarte piet moet een naam van een persoon geheel in chocoladeletters bijelkaar zoeken, alleen moet dat wel precies in de juiste volgorde, omdat het anders verkeerd in het pakje komt. Hieronder is een voorbeeld te zien van de route die zwarte piet moet lopen:



Zwarte piet roept jou hulp in om vanuit zijn beginpunt de kortste route te bepalen. Als piet van een bepaalde doos naar de doos links of rechts er van gaat heeft hij één lengte-eenheid afgelegd.

Omdat de dozen in een cirkel staan is de afstand tussen doos n en doos 1 ook één lengte-eenheid.

Invoer: **Bestand: C.IN**

Het invoerbestand bevat eerst een regel met daarop een getal met het aantal test-sets.

Dan per test-set een regel met het aantal dozen *n*(<=1000) en de lengte van de naam *m*(<=15) en op de volgende regel *n* hoofdletters, die de verschillende letters aangeven. Het eerste karakter is het startpunt van de piet.

De regel erna bevat tenslotte *m* karakters die de naam aangeeft die gezocht moet worden.

Uitvoer: **Bestand: C.OUT**

Het uitvoerbestand bevat voor elke test-set een regel met daarop de lengte van de kortste route vanaf het startpunt totaan de laatste doos.

Voorbeeld:

Bij de invoer: (C.IN)

```
2
8 4
PTABCDIE
PIET
12 2
AEWKEJSDDDDT
ED
```

Hoort de uitvoer: (C.OUT)

```
5
4
```

Beschrijving:

Er was eens een exploitant van een heel groot pretpark. Alles was er zo'n beetje wel te vinden: een schommel, een draaimolen, een achtbaan en zelfs een sprookjestuin.

De beste man had alles goed lopen, maar begon behoefte te krijgen aan een nieuw evenement. Wekenlang piekerde hij om op een origineel idee te komen. Uiteindelijk had hij het: Holdoof, een variant op het bekende verdwaal-vertier.

Holdoof lijkt erg op Doolhof, met enkele bijzondere verschillen. Een loper krijgt bij het begin een maximaal aantal stappen dat hij *per beurt* mag lopen. Hij zet deze passen allemaal in dezelfde richting; als hij een afscheiding tegenkomt, draait hij zich om en loopt de resterende passen de andere kant op.
Daarnaast mogen de speurders niet zomaar kiezen waar ze heen lopen! Op de plaats waar ze stil komen te staan staat een pijl die hen in de nieuwe richting wijst.
Op een plaats waar geen pijl staat mag de loper zelf bepalen welke kant hij kiest. Verder mag een loper er net zoveel beurten over doen als hij wil.

De exploitant geeft opdracht een paar holdoven te ontwerpen. Hij wil echter weten of het eindpunt wel te bereiken is bij een gegeven holdoof en maximum aantal stappen.

Probleem:

Schrijf een programma dat, gegeven een holdoof inclusief pijlen en een maximaal aantal stappen, antwoord geeft op de vraag of het eindpunt bereikbaar is.

Invoer: **Bestand: D.IN**

De eerste regel bevat het aantal runs. Vervolgens komt een regel met het maximaal aantal stappen (<=16) dat in de betreffende beurt gezet mag worden. Er volgen dan twee getallen, die de breedte resp. hoogte (beide <=100) van het hooldof aangeven.
Uiteindelijk volgt de opmaak van het hooldof. Betekenis van de karakters: X – muur/heg; U – pijl omhoog; D – pijl omlaag; L – pijl links; R – pijl rechts; B – beginveld; E – eindveld.
Ieder veld is omgeven door 'X'-en.

Uitvoer: **Bestand: D.OUT**

Per run dient één regel als uitvoer te worden gegeven met 'mogelijk' of 'niet mogelijk' wanneer het eindveld bereikt resp. niet bereikt kan worden.

Voorbeeld:

Bij de invoer: (D.IN)

```
2
2
4 6
XXXX
XBXX
XDLX
XRUX
XXEX
XXXX
4
7 8
XXXXXXX
XDX   X
XUX X X
XLRLRDX
XDXUX X
XULRUUX
XBXEXXX
XXXXXXX
```

hoort de uitvoer: (D.OUT)

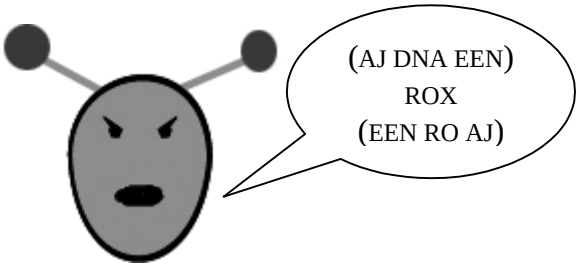
```
niet mogelijk
mogelijk
```

Baaleoonse Expressies — Opgave E

Beschrijving:

Op de planeet Baaleon in het Omega kwadrant huist een intelligente levenssoort. Het vreemde aan deze Baaleons is dat ze een vreemde manier van communiceren hebben. Hun antwoorden op vragen zijn namelijk altijd een soort expressie. Een voorbeeld hiervan is: (AJ DNA EEN) ROX (EEN RO AJ)

Kapitein Tarzanweg heeft haar wetenschaps-



staf de opdracht gegeven de expressies te ontcijferen. Ze zijn daar al een heel eind mee gevorderd. Zo weten ze dat AJ = waar en EEN = niet waar, dat de haakjes net als bij het menselijk ras als groepering werken en dat DNA, ROX en RO operatoren zijn:

AJ DNA AJ	=	AJ
AJ DNA EEN	=	EEN
EEN DNA AJ	=	EEN
EEN DNA EEN	=	EEN
AJ RO AJ	=	AJ
AJ RO EEN	=	AJ
EEN RO AJ	=	AJ
EEN RO EEN	=	EEN
AJ ROX AJ	=	EEN
AJ ROX EEN	=	AJ
EEN ROX AJ	=	AJ
EEN ROX EEN	=	EEN

Verder is het zo dat in een expressie eerst DNA wordt uitgevoerd, dan RO en dan ROX.

Probleem:

Kapitein Tarzanweg wil graag een computerprogramma dat een expressie kan evalueren, zodat ze sneller met de baaleons kan communiceren.

Invoer: **Bestand: E.IN**

Op de eerste regel het aantal test-sets, daarna voor elke test-set een regel met de expressie. De maximum lengte van de expressie is 250 karakters. Alles is in hoofdletters gegeven. Men mag er vanuit gaan dat elke expressie syntactisch correct is.

Uitvoer: **Bestand: E.OUT**

Voor elke test-set een regel met ‘waar’ of ‘niet waar’.

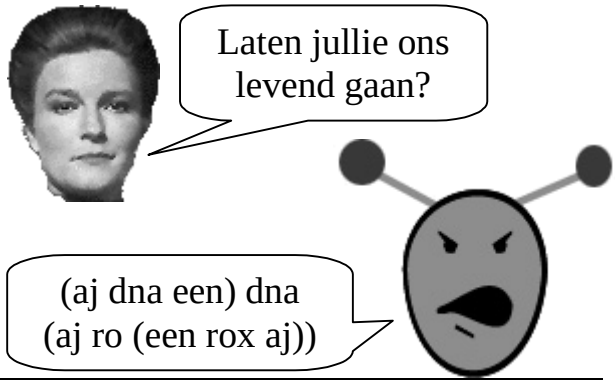
Voorbeeld:

Bij de invoer:

```
2
AJ RO EEN DNA AJ
(AJ DNA EEN) ROX ((EEN))
```

Hoort de uitvoer:

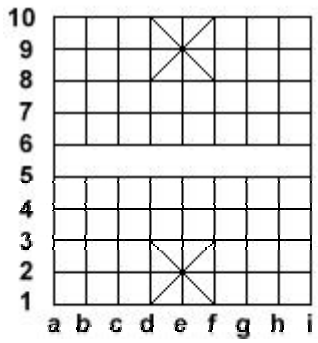
```
waar
niet waar
```



Beschrijving:

Xiangqi (Chinees Schaken) is een variant op het ons wel bekende schaakspel. Het stamt ongeveer uit het begin van de jaartelling en is zeer populair in Zuidoost Azië. Hieronder een overzicht van de regels:

Het Bord



Het bord bestaat uit 10 horizontale lijnen en 9 verticale lijnen. In het midden van het bord zijn de vakken leeg. Dit wordt de rivier genoemd, een scheiding tussen beide 'eigen' kanten. De stukken worden niet zoals met normaal schaken in een vakje gezet, maar op de kruisingen van lijnen.

Zoals je ziet zijn er ook een aantal diagonale lijnen, aan beide kanten elk een gebied van 3 bij 3. Dit gebied is het paleis. De generaal en zijn adviseurs mogen dit gebied niet uit.

Stukken:

De generaal (J)

De generaal mag horizontaal en verticaal één stap verzetten, zolang deze maar binnen het paleis blijft. Het doel van het spel is om de generaal van de tegenpartij te slaan.
Een Generaal mag nooit rechtstreeks naar de andere Generaal 'kijken'. D.w.z.: ze mogen nooit op één lijn staan zonder een stuk van één van beide partijen ertussen.

De adviseur (S)

De twee adviseurs aan elke kant zijn er om de generaal te beschermen. Zij mogen dan ook net zoals de generaal het paleis niet uit. Verplaatsen en stukken slaan doen zij echter alleen diagonaal, dus over de diagonale lijnen die in het bord getekend zijn.

Olifant (X)

De olifant is een verdedigend stuk, omdat het vanwege zijn gewicht niet de rivier overkan. De olifant beweegt diagonaal steeds met precies twee stappen. Hierbij mag er geen ander stuk in de weg staan.

Strijdwagen (C)

De strijdwagen is vergelijkbaar met de toren uit normaal schaken. Horizontaal en verticaal een onbeperkt aantal stappen (zonder natuurlijk over andere stukken heen te gaan).

Paard (M)

Een paard beweegt net als in gewoon schaken eerst twee velden horizontaal en dan één verticaal of eerst twee velden verticaal en dan één veld horizontaal. De laatste twee stappen kun je ook zien als een diagonale stap.
Het paard heeft als handicap t.o.v. normaal schaak dat het niet over stukken heen kan springen. Dus de positie na de eerste stap moet onbezet zijn. (De tweede stap is diagonaal en dus de eindpositie).

Kanon (P)

Het kanon beweegt net als de strijdwagen, maar stukken slaan gaat iets anders. Om een stuk te slaan moet het kanon over één ander stuk bewegen, wel nog steeds alleen horizontaal of verticaal. Er moet dus een willekeurig stuk van één van beide partijen staan tussen het kanon en het stuk dat geslagen wordt.

Soldaat (Z)

De soldaat mag in het 'eigen gebied' alleen maar één stap vooruit bewegen. Zodra het de rivier over is, is het stuk krachtiger. Terugtrekken mag nog steeds niet, maar één stap naar links of rechts is ook mogelijk.

Probleem:

Gevraagd is om in een bepaalde stelling alle mogelijke zetten te genereren die voldoen aan de voorwaarden in het voorgaande.

Invoer: **Bestand: F.IN**

Eerst een regel dat het aantal runs aangeeft, daarna voor elke run een regel met een 'R' als rood aan slag is en een 'B' als blauw aan slag is. Daarna een matrix van 10 bij 9 met daarin een '.' als er geen stuk en verder worden hoofdletters voor rode stukken en kleine letters voor blauwe letters gebruikt.

Uitvoer: **Bestand: F.OUT**

De uitvoer bestaat voor elke testset eerst uit een regel met het nummer van de testset. Daarna voor elke zet die gevonden is een regel met de letter die het stuk aangeeft, een spatie, de beginpositie, een streepje en de positie waar het stuk terecht komt. Als een stuk geslagen wordt, komt daarna nog een spatie en de letter die aangeeft welk stuk geslagen wordt.

De zetten moeten eerst op stuk gesorteerd worden (alfabetisch), daarna op beginpositie.

Voorbeeld:

Bij de invoer: (Zie voorbeeld A.IN)

```
1
R
...j.....
.........
.........
....Z....
.........
.....z...
.....Z...
.........
.........
...SJS...
```

Hoort de uitvoer: (A.OUT)

```
1
J E1-E2
S D1-E2
S F1-E2
Z E7-D7
Z E7-E8
Z E7-F7
Z F4-F5 z
```

Beschrijving:

Bill Poorts heeft plannen voor een nieuw software produkt: Vensters 2001. Hij heeft zijn marketingafdeling al aan het werk gezet en inmiddels draaien er al grootscheepse reclamecampagnes voor dit produkt.

Er is echter nog één probleempje: hij moet nog beginnen met programmeren. Nu hoeft dat geen probleem te zijn, ware het niet dat hij net 99,99% van zijn programmeerafdeling ontslagen heeft, omdat hij meer marketing mensen en juristen nodig had.

Oftewel: hij heeft nog maar één programmeur over. Deze programmeur is echter niet al te slim, hij begrijpt namelijk de source code die er al ligt van Vensters 98 niet. Daarom besluit hij maar helemaal opnieuw te beginnen.

Een van de eerste problemen die hij tegenkomt heeft zit in de muisroutines. Hij moet van een bepaalde muisklik bepalen in welk venster die thuishoort. Nu is dat niet al te moeilijk, behalve als er vensters over elkaar liggen, dan moet dat venster dat op de voorgrond staat natuurlijk het resultaat zijn.

Probleem:

Gegeven een groot aantal rechthoekige vensters, waarbij de eerste het meest vooraan staat en het laatste helemaal onderop. Bepaal voor een aantal muiskliks in welk venster deze thuishoren.

Het scherm heeft een grootte van maximaal 50000 bij 50000 pixels.

Invoer: **Bestand: G.IN**

Eerst een regel met het aantal test-sets. Vervolgens voor elke test-set een getal *n*, het aantal vensters. Dan *n* keer een regel met vier getallen, die het x en y coordinaat van de pixel linksboven en rechtsonder aangeven. Op de regel erna een getal *m*, het aantal muiskliks. Vervolgens *m* keer een regel met een x- en y-coordinaat van een muisklik.

Uitvoer: **Bestand: G.OUT**

Voor elke muisklik het nummer (tussen 1 en *n*) van het venster waar het in thuis hoort. Als het in geen enkel venster thuis hoort, moet de tekst ‘bureaublad’ afgedrukt worden.

Voorbeeld:

Bij de invoer: (G.IN)

```
2
4
0 0 100 100
50 50 150 150
100 100 200 200
150 150 250 250
2
10 10
60 120
1
0 0 1 1
1
2 2
```

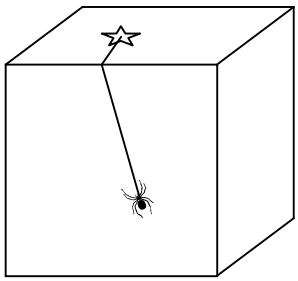
Hoort de uitvoer: (G.OUT)

```
1
2
bureaublad
```

Beschrijving:

In een kubusvormige kamer zit een spin op de muur. Dan verschijnt er plotseling een lekkere vette vlieg in de kamer en die gaat ergens op het plafond zitten.

Omdat de spin zelf niet kan vliegen en die vlieg er toch wel heel erg lekker uitziet probeert hij een manier te vinden om zo snel mogelijk bij zijn lekkere hapje te komen.



Probleem:

Gegeven de positie van de spin en de positie van de prooi, de korste afstand van de spin tot de prooi over de muur.

Dit is dus altijd een rechte lijn. Het antwoord moet bovendien in wortelnotatie gegeven worden.

Invoer: **Bestand: H.IN**

Het invoerbestand bevat eerst een regel met het aantal test-sets. Dan volgen per test-set5 drie regels:

1. Een getal: *n* de afmeting van de kubus
2. Drie gehele getallen tussen 0 en n, die het x, y en z coordinaat van de spin aangeven
3. Drie gehele getallen tussen 0 en n, die het x, y en z coordinaat van de prooi aangeven

Men mag er vanuit gaan dat beide coordinaten zich op een zijde van de kubus bevinden.

Uitvoer: **Bestand: H.OUT**

De uitvoer bevat voor elke test-set de kortste afstand in wortel notatie. Deze wortel moet natuurlijk wel zoveel mogelijk vereenvoudigd worden, dus √18 = 3√2.

De wortelnotatie bevat twee regels, op de tweede regel wordt het wortelteken gemaakt door een backslash en een slash '\/'. Dan volgt het getal. De eerst regel bevat underscores boven het getal zelf.

Mocht het antwoord een geheel getal zijn, dan is de eerste regel leeg en bevat de tweede regel het getal.

Voorbeeld:

Bij de invoer:

```
3
8
4 0 7
5 8 7
5
5 0 0
0 0 0
10
0 2 4
6 6 10
```

hoort de uitvoer:

		_	_	_
\	/	1	0	1
5				
			_	_
4	\	/	1	0

Oftewel in gewone tekst:

```
 ___
\/101

5

  __
4\/10
```