Eerste ronde Nederlandse Informatica Olympiade 2015 -2016

De informatica olympiade is een wedstrijd voor leerlingen uit het voortgezet onderwijs in Nederland. Het is een wedstrijd die bestaat uit drie ronden. In de derde ronde wordt bepaald wie Nederland mogen vertegenwoordigen op de Internationale Informatica Olympiade in juli 2016 in Rusland.

De eerste ronde

De eerste ronde van de Nederlandse Informatica Olympiade bestaat dit jaar uit 12 opgaven. Die hoef je niet allemaal te maken, al mag dat natuurlijk wel. Deelnemers die tenminste 200 punten halen krijgen een certificaat.

Heb je tussen de 200 en 399 punten dan staat op het Certificaat de vermelding **Brons**, tussen de 400 en 599 punten de vermelding **Zilver** en bij 600 punten of meer punten de vermelding **Goud**.

Soort	Omschrijving	Aantal	Punten per opgave	Totaal te behalen
A	Inleidende opgaven	5	40	200
B	Theoretische opgaven	4	50	200
C	Programmeeropgaven	2	100	200
D	Een spel programmeren	1	100	100

De beste 100 leerlingen worden uitgenodigd voor de tweede ronde, die in maart 2016 wordt gehouden op de Universiteit Twente. Voor deelname aan die tweede ronde moet je wel minstens 200 punten hebben gehaald.

Voor de beste deelnemer van iedere klas is een aparte prijs beschikbaar.

Om deel te kunnen nemen moet je een account maken op submit.informaticaolympiade.nl

Bij de eerste keer aanmelden moet je enkele gegevens aanleveren, die wij nodig hebben om de olympiade goed te kunnen organiseren. Als je deze gegevens niet wilt of kunt aanleveren, kun je helaas niet deelnemen. Je verklaart in de laatste stap dat je de gegevens naar waarheid hebt ingevuld; daarna staat deelname voor je open. Als je van vorige jaren al een account hebt, zul je de gegevens ook eventueel eerst moeten aanvullen voor je verder kunt werken in het systeem.

Je kunt je uitwerkingen uploaden naar submit.informaticaolympiade.nl wanneer je in het systeem bent ingelogd. In het systeem kun je ook een voorbeeldopgave insturen om uit te proberen hoe het werkt. De opgaven worden meteen geheel of gedeeltelijk nagekeken, voor de rest van de uitslag zul je moeten wachten op het resultaat. Je uitwerkingen voor de opgaven A, B en C moeten uiterlijk 20

januari worden geupload. Op 23 januari wordt de eerste ronde gejureerd en kort daarna worden de uitslagen gepubliceerd.

Voor de spelopgave, opgave D, moet je je aanmelden op nio3.codecup.nl en kun je via die site ook je programma uploaden. De deelnemende programma's die meewerken met het jurysysteem komen op 23 januari 2016 tegen elkaar uit in een toernooi dat te volgen is op nio3.codecup.nl en de winnaar wint de jaarlijkse Windesheim Digitalisprijs van 200 euro.

Voor alle opgaven geldt dat je er van uit mag gaan dat je programma's alleen correcte invoer aangeboden krijgen. Tenzij anders aangegeven krijgt je programma één seconde om een testinvoer op te lossen.

Toelichting opgaven

Opgaven A1 tot en met A5

Deze opgaven zijn vooral bedoeld voor leerlingen die beginnen met programmeren. Vanuit de olympiade bieden we lesmateriaal aan om te beginnen met programmeren met Python. Dat is de cursus CS Circles van de Universiteit van Waterloo in Canada. Er is een Nederlandse vertaling beschikbaar; zie http://cscircles.cemc.uwaterloo.ca/0-nl/. In de tekst van die Nederlandse vertaling zal worden aangegeven wanneer je toe bent aan de volgende opgave van de eerste ronde.

Opgaven B1 tot en met B4

Deze opgave kun je één voor één downloaden uit het inzendsysteem. De opgave wordt speciaal voor jou gemaakt en jij moet het antwoord op de opgave die je vanuit het systeem krijgt inleveren. Het heeft dus geen zin om de antwoorden van iemand anders te gebruiken en die in te zenden.

Als je binnen een week het goede antwoord instuurt krijg je 50 punten per opgave. Voor iedere dag later gaat er één punt van je score af. Inzendingen na 20 januari 2016 zullen niet worden verwerkt.

Als je een verkeerd antwoord hebt gegeven, verlies je 10 punten. Ja mag wel weer een nieuwe poging wagen.

Het gaat bij al deze opgaven om korte antwoorden, een getal of een korte tekst, die je op de betreffende pagina van het inzendsysteem kunt invoeren. Als je je antwoord hebt bevestigd, krijg je meteen je score te zien.

Je mag allerlei hulpmiddelen gebruiken om de opgave op te lossen. Je zou er bijvoorbeeld een computerprogramma bij kunnen schrijven. Noodzakelijk is dat echter niet. Als voorbereiding op het vervolg van de informatica olympiade is het wel een mooie uitdaging om na te gaan hoe je een programma zou kunnen schrijven dat dit probleem, of problemen die er op lijken, kunt oplossen.

Opgaven C1 en C2

Dit zijn wat complexere opgaven waarmee je een probleem moet oplossen door het schrijven van een computerprogramma. Die programma's lezen invoer van standard input (het toetsenbord) en schrijven uitvoer naar standard output (het beeldscherm). Je programma moet zich daarbij precies

houden aan de beschrijvingen van de opdracht. Je programma krijgt een aantal testgevallen voorgeschoteld en voor ieder testgeval kun je punten krijgen.

Opgave D

Bij deze opgave moet je een programma schrijven dat het spel 2187 kan spelen. De programma's spelen op 23 januari een toernooi tegen elkaar. Om deel te kunnen nemen moet je programma kunnen samenwerken met onze jurysoftware; voor details verwijzen we naar nio3.codecup.nl

De CodeCup

De Stichting Nederlandse Informatica Olympiade organiseert nog een wedstrijd waarbij een spel moet worden geprogrammeerd. Daaraan mag iedereen deelnemen; de afgelopen jaren hebben we deelnemers gehad uit meer dan twintig verschillende landen. Zie voor deze wedstrijd www.codecup.nl

Opgave A1. Halter

Schrijf een programma dat een getal N inleest van standard input. Het programma schrijft naar standard output een “halter” van 2N-1 regels van elk 2N-1 tekens, in een patroon zoals te zien is in het voorbeeld hieronder. De eerste N regels vormen een driehoekig patroon, de laatste N ook.

Voorbeeld

Invoer: 7

Uitvoer:

```
*************
_***********_
__*********__
___*******___
____*****____
_____***_____
______*______
_____***_____
____*****____
___*******___
__*********__
_***********_
*************
```

Randvoorwaarde: N is een positief geheel getal, met 1 < N < 41

Opgave A2. Naamcode

Schrijf een programma dat een naam inleest van standard input. De eerste letter van alle voornamen en de achternaam is telkens een hoofdletter, de andere letters zijn kleine letters. Tussenvoegsels zijn kleine letters. Dubbele achternamen komen niet voor, de achternaam is dus altijd één woord.

Het programma schrijft één regel uitvoer naar standard output. Dat is de naamcode die bij de betreffende naam hoort. Zo'n naamcode bestaat uit drie hoofdletters. Eerst de eerste letter van de achternaam, dan de laatste letter van de achternaam, tenslotte de eerste letter van de voornaam.

Voorbeeld:

Invoer: `Anne Louise van der Boom`

Uitvoer: `BMA`

Randvoorwaarde: In de invoer staan maximaal 60 tekens.

Opgave A3. Zevenachtig

Schrijf een programma dat van standard input gehele niet-negatieve getallen inleest, tot er een 0 wordt ingelezen. Het programma schrijft één regel uitvoer naar standard output; daarop staat het aantal zevenachtige getallen in de invoer. Een getal is zevenachtig als het voldoet aan minstens één van de volgende eigenschappen:

1. Het is een positief geheel getal dat deelbaar is door 7.
2. Het is een getal waar het cijfer 7 minstens één keer in voorkomt.

Voorbeelden van zevenachtige getallen zijn 14, 17 en 77.

Voorbeeld

Invoer:
```
21
7
66
37
0
```

Uitvoer:
```
3
```

Randvoorwaarde: De invoer bestaat uit maximaal 100 getallen. Elk getal in de invoer is niet groter dan 100.

Opgave A4. Foto's selecteren

Je hebt een grote stapel foto's, die je allemaal al eens gewaardeerd hebt met een uniek getal; hoe groter het getal, des te meer waarde ken je toe aan de foto.

Je wilt een selectie maken van de foto's. Dat doe je als volgt. Je legt ze op een grote stapel, waarna je de eerste tien afpakt en op de volgorde waarin je ze afpakt voor je neer legt. De hoogst gewaardeerde foto kies je uit. Alle foto's die boven de gekozen foto lagen leg je aan de kant. Je vult je rij aan met foto's tot je er weer tien hebt, of tot de grote stapel op is. Dan herhaal je bovenstaande procedure. Als de stapel op is, kies je voor de laatste keer een foto die je wilt selecteren, behalve als er dan geen foto's meer voor je liggen natuurlijk.

Schrijf een programma dat een aantal regels inleest van standard input.

- Op de eerste regel staat het aantal F aan foto's op je stapel.
- Op de volgende F regels staat de unieke waardering van een foto. Dat is een positief geheel getal, niet groter dan F.

Je programma schrijft voor iedere geselecteerde foto één regel uitvoer naar standard output, met daarop de waardering van de geselecteerde foto.

Voorbeeld

Invoer:

```
22
9
12
17
22
8
1
20
3
15
21
2
19
4
18
5
16
14
13
11
7
6
10
```

Uitvoer:
```
22
21
19
18
```

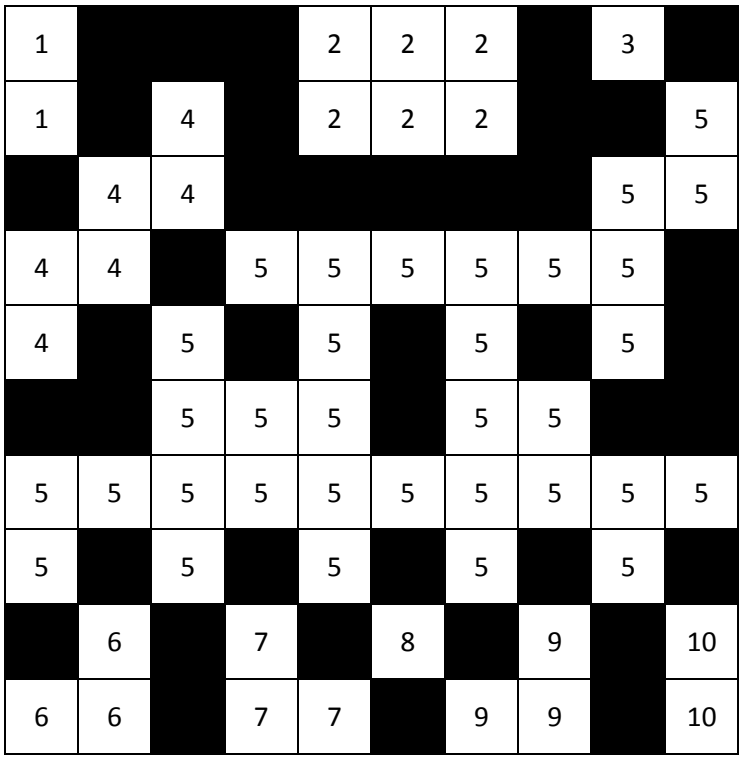
Toelichting bij het voorbeeld:

9	12	17	22	8	1	20	3	15	21	2	19	4	18	5	16	14	13	11	7	6	10
9	12	17	22	8	1	20	3	15	21	2	19	4	18	5	16	14	13	11	7	6	10
9	12	17	22	8	1	20	3	15	21	2	19	4	18	5	16	14	13	11	7	6	10
9	12	17	22	8	1	20	3	15	21	2	19	4	18	5	16	14	13	11	7	6	10

- In de eerste ronde wordt foto 22 gekozen. De foto's 9, 12 en 17 worden weggelegd. Er worden vier nieuwe foto's van de stapel gepakt.
- In de tweede ronde wordt foto 21 gekozen. De foto's 8, 1, 20, 3 en 15 worden weggelegd. Er worden zes nieuwe foto's van de stapel gepakt.
- In de derde ronde wordt foto 19 gekozen. Foto 2 wordt weggelegd. Er worden twee nieuwe foto's van de stapel gepakt.
- In de vierde ronde wordt foto 18 gekozen. Foto 4 wordt weggelegd. Er kunnen geen nieuwe foto's meer van de stapel worden gepakt, want die is leeg. Daarom stopt het selectieproces.

Randvoorwaarden: Er geldt 20 < F < 1000.

Opgave A5. Gebieden



Deze opgave gaat over een soort dambord van witte en zwarte vakjes. Witte vakjes die met een zijde aan een ander wit vakje grenzen maken deel uit van hetzelfde gebied. In het plaatjes hierboven zijn 10 verschillende gebieden van witte vakjes; je kunt niet van het ene witte gebied naar het andere komen door horizontaal of verticaal één stapje te doen, zonder dat je over een zwart vakje komt.

Schrijf een programma dat tien regels van tien tekens inleest van standard input. Iedere regel bestaat uitsluitend uit de tekens 0 en 1. Een 0 staat voor een wit vakje, een 1 voor een zwart. Het dambord heeft een grootte van 10x10 vakjes.

Je programma telt het aantal aaneengesloten gebieden van witte vakjes in de invoer.

Voorbeeld

Invoer:
```
0111000101
0101000110
1001111100
0010000001
0101010101
1100010011
0000000000
0101010101
1010101010
0010010010
```

Uitvoer:
```
10
```

Opgave B1 tot en met B4:

Download deze van http://submit.informaticaolympiade.nl

B1. Mobiele telefoniemast

B2. Verbindingswegen

B3. Langste oplopende deelrij

B4. Woorden leggen

Opgave C1. Delercode

De delercode is een manier om de tekens van een tekst een andere volgorde te geven, zodat een argeloze lezer niet weet wat er staat (behalve natuurlijk als hij of zij snapt hoe de delercode werkt). Stel de tekst bestaat uit N tekens. Voor de getallen G van 2 tot en met N doe je nu het volgende:

- Is het getal G een deler van N, dan schrijf je eerst de tekens op de posities die een veelvoud van G zijn op. Daarna neem je alle tekens die je daarbij had overgeslagen. Met het nu verkregen 'woord' werk je vervolgens verder. Daarna ga je door naar het volgende getal.
- Is G geen deler van N, dan sla je het getal over.

Een voorbeeld:

Woord	M	E	D	A	I	L	L	E	S	P	I	E	G	E	L
G=3	D	L	S	E	L	M	E	A	I	L	E	P	I	G	E
G=5	L	L	E	D	L	S	E	M	E	A	I	E	P	I	G
G=15	G	L	L	E	D	L	S	E	M	E	A	I	E	P	I

Het oorspronkelijke woord is `MEDAILLESPIEGEL`. Dit woord heeft een lengte van 15 en heeft als delers respectievelijk 3, 5 en 15. Bij G=3 worden eerst de tekens op de plaatsen 3, 6, 9 etc. overgenomen in het nieuwe woord, daarna de overgebleven tekens van de overgeslagen plaatsen 1, 2, 4 en zo verder. Als G=5 gebeurt met het in de voorgaande stap verkregen woord hetzelfde, maar nu met de tekens op de plaatsen 5, 10 en 15. En als G=15 wordt het teken op plaats 15 naar voren gehaald. Het resultaat van het toepassen van deze delercode levert uiteindelijk het woord `GLLEDLSEMEAIEPI` op (verkregen op de manier zoals in bovenstaande tabel te zien).

In deze opgave krijg je een tekst waarop de delercode is gebruikt. Jouw programma moet bepalen hoe de oorspronkelijke tekst er uit ziet.

Schrijf een programma dat invoer leest van standard input. De invoer is een tekst van minstens vijf, hoogstens zeventig tekens. Er komen geen spaties in de invoer voor.

Je programma schrijft als uitvoer een regel tekst naar standard output. Als op deze tekst de delercode wordt toegepast moet dat de tekst van de invoer als resultaat hebben.

Voorbeeld

Invoer: `GLLEDLSEMEAIEPI`

Uitvoer: `MEDAILLESPIEGEL`

Er geldt voor deze opgave een tijdlimiet van 2 seconden.

Opgave C2. Sorteren door omkeren

In deze opgave krijgt je programma als invoer een rij met getallen. Deze getallen staan niet op volgorde. Je mag één type bewerking `omkeren` gebruiken om de getallen in oplopende volgorde te krijgen: Je wijst een begin- en eindpositie aan en je wisselt de getallen op en tussen deze posities om.

Een voorbeeld. Beginsituatie:

Getal	5	20	4	22	23	26	27	1	14	21
Positie	1	2	3	4	5	6	7	8	9	10

`omkeren(7,10)` Wissel de getallen op de posities 7 tot en met 10 om.

Getal	5	20	4	22	23	26	21	14	1	27
Positie	1	2	3	4	5	6	7	8	9	10

Het gaat er nu om dat je programma een manier vindt om de gegeven rij met zo weinig mogelijk aanroepen van deze bewerking `omkeren` gesorteerd te krijgen.

Schrijf een programma dat twee regels invoer leest van standard input.

- Op de eerste regel staat het aantal getallen N in de rij ($3 < N < 16$).
- Op de tweede regel staan N verschillende positieve gehele getallen (elk kleiner dan 100), die aan het begin van het programma in die volgorde op de posities 1 tot en met N staan, telkens gescheiden door een spatie.

Je programma voert een aantal regels uit naar standard output. Op het eerste vakje staat het aantal keren B dat je programma de bewerking `omkeren` nodig heeft. Op de volgende B regels staan telkens twee getallen, de begin- en eindpositie van het deel van de rij dat wordt omgekeerd. Na deze B bewerkingen moeten de getallen in de rij oplopend zijn gesorteerd. Bovendien moet B bij voorkeur zo klein mogelijk zijn.

Voorbeeld

Invoer:
```
10
5 20 4 22 23 26 27 1 14 21
```

Uitvoer:
```
7
7 10
6 9
5 8
4 7
2 5
2 4
1 3
```

Beginsituatie	5	20	4	22	23	26	27	1	14	21
`omkeren(7,10)`	5	20	4	22	23	26	21	14	1	27
`omkeren(6,9)`	5	20	4	22	23	1	14	21	26	27
`omkeren(5,8)`	5	20	4	22	21	14	1	23	26	27
`omkeren(4,7)`	5	20	4	1	14	21	22	23	26	27
`omkeren(2,5)`	5	14	1	4	20	21	22	23	26	27
`omkeren(2,4)`	5	4	1	14	20	21	22	23	26	27
`omkeren(1,3)`	1	4	5	14	20	21	22	23	26	27
Positie	1	2	3	4	5	6	7	8	9	10

Let op: Dit is NIET de beste oplossing bij deze invoer!

Bij veel invoergevallen zijn er verschillende oplossingen mogelijk. Je programma moet één oplossing geven.

Voor deze opgave geldt een tijdlimiet van 2 seconden.

In de helft van de testgevallen bestaat de rij uit minder dan 10 getallen.

Voor elk testgeval geldt dat je (N - <jouw_antwoord>)/(N - <beste_antwoord>)*10 punten krijgt, als je een geldige reeks bewerkingen hebt uitgevoerd; er zijn in totaal 10 testcases.

Opgave D. 2187

Het spel 2187 is een variant op het welbekende spel 2048. Dit spel wordt met één speler gespeeld, terwijl 2187 met twee spelers gespeeld wordt. De twee spelers noemen we A en B en ze spelen om de beurt, waarbij A mag beginnen. Ze leggen volgens een vast protocol (zie hieronder) soms een steen op het bord en schuiven soms de stenen op het bord naar een zijkant, zoals ook gebeurt in het spel 2048. Er worden geen willekeurige stenen met een 2 of met een 4 geplaatst: de spelers plaatsen zélf stenen met waarde 1 op het bord.

Het doel van het spel is om samen zoveel mogelijk punten op het bord te krijgen. De score is gelijk aan de som van de waarde van de stenen. Het maximum aantal punten dat jullie samen op het bord kunnen krijgen, wordt jullie gezamenlijke score. Het maakt niet uit of deze maximum score onderweg op het bord komt, of dat deze tegen het einde wordt bereikt.

Het gebruikte bord is 4 bij 4 hokjes en is bij het begin van het spel leeg.

Zettenvolgorde – Het protocol

In 2187 bestaan niet alleen stenen met verschillende getallen erop, maar hebben de stenen ook twee verschillende kleuren: blauw en rood. In andere spellen zijn alle zetten qua type hetzelfde, maar omdat dit hier niet het geval is, is het nodig dat duidelijk is vastgesteld in welke volgorde welke zetten gedaan worden. Zie hiervoor de volgende tabel:

	Speler	**Actie**
1	A	Plaats een blauwe steen
2	B	Plaats een rode steen
3	A	Plaats een rode steen
4	B	Plaats een blauwe steen
5	A	Schuif het bord
6	B	Schuif het bord

Deze zes acties herhalen zich tot het spel afgelopen is. Het spel is afgelopen als er op een gegeven moment geen geldige zet meer gedaan kan worden, of wanneer er in totaal 1000 zetten zijn gedaan.

Spelmechaniek

Een speler kan een steen **plaatsen** op een willekeurig leeg vakje. Als dit niet meer kan omdat het bord vol is, is het spel afgelopen. Als een speler een steen moet plaatsen, moet die de coördinaten van het doelvakje als twee cijfers uitvoeren, zonder spatie ertussen, met eerst de rij, de horizontale coördinaat en dan de kolom, de verticale. In het volgende plaatje liggen er stenen op de posities 11, 14, 33 en 42.

Als een speler het bord moet **schuiven**, heeft die vier opties. Er kan vier kanten op geschoven worden: naar boven (U), naar rechts (R), naar links (L) en naar beneden (D). Schuifzetten worden aangeduid met één hoofdletter.

Stel, als voorbeeld, dat de speler naar boven schuift in het volgende bord. Aan de hand van dit voorbeeld wordt uitgelegd hoe de schuiflogica werkt.

Voor het schuiven

Na het schuiven (naar boven)

In de linkerkolom staan drie stenen: twee rode stenen met een 3 en daaronder een blauwe 3. Bij het naar boven schuiven gaan die twee rode stenen samen tot een 9 (de volgende macht van 3) omdat ze hetzelfde getal en dezelfde kleur hebben. De blauwe 3 blijft dan over en schuift er onder tegenaan. Resultaat is een rode 9 met daaronder een blauwe 3.

In de tweede kolom staan een rode 1, een blauwe 1, nog een blauwe 1 en een rode 1. Van bovenaf bekeken (we schuiven immers naar boven) zien we eerst twee stenen met hetzelfde getal maar met *verschillende* kleuren. Om deze reden gaan de stenen niet samen, maar verdwijnen ze juist allebei. Hetzelfde geldt voor de blauwe 1 en de rode 1 die eronder staan: gelijke getallen zorgen ervoor dat ze met elkaar reageren, maar verschillende kleuren zorgen dat ze niet samen gaan tot de volgende macht van 3, maar elkaar vernietigen. In de eerste kolom was te zien dat als de kleuren juist hetzelfde zijn, de stenen *wel* samengaan tot de volgende macht van 3.

In de derde kolom staan twee blauwe stenen met een 9. Bij het naar boven schuiven gaan deze tegen elkaar aan en omdat het getal *en* de kleur gelijk zijn, gaan ze samen tot de volgende macht van 3.

3^0	3^1	3^2	3^3	3^4	3^5	3^6	3^7
1	3	9	27	81	243	729	2187

In de laatste kolom zien we van bovenaf eerst een rode 9 en een blauwe 3. Deze reageren niet met elkaar. Als volgende mogelijkheid zien we een blauwe 3 en een blauwe 1. Deze reageren ook niet met elkaar. Als volgende mogelijkheid zien we een blauwe 1 en nog een blauwe 1. Deze getallen zijn wel gelijk, en omdat de kleuren ook gelijk zijn, gaan deze samen tot een blauwe 3. Door dit samengaan komen er twee blauwe stenen met een 3 boven elkaar, maar deze reageren niet meer met elkaar in deze schuifactie. Deze zouden pas reageren als hierna *nog* een keer naar boven geschoven zou worden.

In het kort is de mechaniek van het schuiven dus als volgt. Je kijkt vanaf de kant waar je naartoe schuift in volgorde naar de stenen die je tegenkomt. Als twee opeenvolgende stenen een gelijk getal hebben, reageren ze met elkaar. Als ze dan ook nog dezelfde kleur hebben, gaan ze samen tot de volgende macht van 3, anders vernietigen ze elkaar. Als de stenen *niet* dezelfde getallen hebben, reageren ze niet. Daarna kijk je naar de stenen die dan volgen. Aan het eind schuif je alle resterende stenen naar desbetreffende kant.

Als er als gevolg van een schuifactie geen enkele verandering optreedt, is die schuifactie ongeldig. Als er een schuifzet aan de beurt is en er is geen geldige schuifactie (als het bord óf leeg óf vol is zonder naastgelegen gelijke getallen), ook dan is het spel afgelopen.

Opgave:

Jij moet een programma schrijven dat een zet van je medespeler ontvangt via de jurysoftware en dat vervolgens een zet teruggeeft aan de jurysoftware. Het is aan jou de taak om aan de hand van de laatste informatie een zo goed mogelijke zet te bepalen!

De zetten van de jurysoftware moet je lezen vanaf stdin (standaard input) en je eigen zetten moet je afdrukken naar stdout (standaard uitvoer).

Als eerste actie moet je programma er achter komen of je als A of als B speelt. De eerste regel die je programma inleest bestaat uit een ‘A’ of een ‘B’. Als je programma ‘A’ inleest dan speel je als A en dan wordt jouw eerste zet als reactie verwacht. In het andere geval speel je dus als B en is de

volgende string die je gekregen hebt de eerste zet van speler A. Je mag er vanuit gaan dat die zet altijd correct is. Dan wordt van jou de eerste tegenzet verwacht. Zo gaat dat door tot het spel is afgelopen.

Als je programma een ongeldige zet doet, of als je programma crasht of er te lang over doet, dan wordt je programma afgesloten. In dat geval neemt de jurysoftware jouw taak over zodat het andere programma het spelletje kan uitspelen. De jurysoftware zal sportief meespelen om je tegenstander de gelegenheid te geven om een goede score te kunnen halen. Als je programma "Quit" inleest in plaats van een gewone zet, dan betekent het dat het spel is afgelopen en dat je jouw programma op de normale manier moet beëindigen.

Als je een zet doorgeeft aan de jurysoftware moet je de string verplicht afsluiten met een newline. Ook ben je verplicht om je uitvoer daarna te flushen zodat de jurysoftware niet gaat zitten wachten totdat de computer de zet heeft doorgegeven aan de jurysoftware (Zie de uitleg bij de technische regels)

Met alle ingezonden programma's die in staat zijn samen te werken met onze jurysoftware wordt een toernooi georganiseerd.

Beide programma's krijgen na afloop van een spel de hoogste score die op het bord gelegen heeft tijdens het spel. Als jouw programma tijdens een spelletje wordt gediskwalificeerd, krijg je echter 0 punten. De andere speler krijgt het aantal punten zoals dat in de regels hierboven is uitgelegd.

Voorbeeld van de communicatie met de jurysoftware:

Speler	**Invoer**	**Uitvoer**
A	A	
B	B	
A		11
B	11	23
A	23	44
B	44	21
A	21	U
B	U	R

Als het spel is afgelopen krijgen beide spelers een "Quit" toegestuurd, en hebben ze de gelegenheid hun programma netjes af te sluiten.

Je programma heeft 30 seconden de tijd om een spel te spelen. De tijd die de tegenstander heeft wordt niet meegeteld.

Deelnemen?

Meld je aan op nio3.codecup.nl en lees daar alles over deze opgave. Je kunt als je bent ingelogd je programma inzenden en de voorrondes bekijken om te zien hoe je programma zich houdt. In de technische regels staat aangegeven waar je programma aan moet voldoen.

De winnaar krijgt de Windesheim Digitalisprijs, een geldbedrag van 200 euro.

Als je programma wordt geaccepteerd voor deelname aan het toernooi verdien je 20 punten voor deze opgave. Als je programma zonder fouten speelt kun je daarmee nog eens 50 punten verdienen. De uitslag van de competitie is bepalend voor de laatste 30 punten.