Opgaven WISB256 midterm

18 mei 2015

Vragen

In dit boekje staan een aantal programmeeropgaven van oplopende moeilijkheidsgraad (1-5). De minimale eis voor een voldoende is dat je de niveau 1 en 2 opgaven goed hebt; maar we verwachten dat niveau 3 ook voor iedereen goed te doen zal zijn. Als je ook de moeilijkere maakt kun je een hoger cijfer krijgen. Merk op dat er vrij veel opgaven in dit boekje staan; het is niet erg als je ze niet allemaal afkrijgt.

Je maakt de opgaven zelf en je kunt aan de hand van de voorbeeld in- en output die is vermeld bij iedere opgave controleren of je programma correct werkt. Als je klaar bent met je programma, kun je hem uploaden naar DOMJudge en zien of je programma inderdaad goed is. Zo ja, ga door met de volgende opgave. Zo niet, probeer het nog eens; je hebt een ongelimiteerd aantal pogingen. Voor meer uitleg over DOMJudge, zie de volgende sectie.

Bij sommige opgaven bestaat de invoer uit maar één regel. In dat geval stelt elke rij in de invoer/uitvoer-tabel een apart testgeval voor. Als de invoer uit meerdere regels bestaat, staat elk testgeval in een apart tabelletje.

Je mag je *eigen* GitHub repository, Cloud9 workspace, aantekeningen, de officiele Python documentatie website `https://www.python.org/doc/` en het boek *Think Python* raadplegen. Het gebruik van andere bronnen is *niet* toegestaan.

Tips

- Begin met de makkelijkste opgave en ga dan door met de moeilijkere.
- Doet je programma er meer dan een paar seconden over, dan is het het waarschijnlijk fout!
- In- en uitvoer gaan met `input()` en `print()`. Moet je meerdere variabelen van één regel lezen, gebruik dan bijvoorbeeld `input().split()` om een string op te splitsen.
- Print *geen* onnodige teksten, zoals "`Geef invoer:`", of "`Antwoord:`", maar print alleen wat er in de opgave vermeld staat. Als jouw uitvoer niet exact overeenkomt, heb je de opdracht fout!
- Bestaat de input uit een vast aantal regels, gebruik dan meerdere `input` commando's.
- Soms moet je een variabel aantal regels inlezen waarbij de eerste input het aantal regels geeft, lees dan eerst het aantal in en gebruik vervolgens een `for` loop om de rest van de input regel voor regel in te lezen.

Succes!

DOMJudge

Het inzenden van opgaven kan via DOMJudge (`https://domjudge.cs.uu.nl/wisb256/team/`). Om een oplossing in te zenden kies je één bestand, de letter van de opgave, en de taal (Python 3). Merk op dat het niet mogelijk is om een oplossing in te sturen die uit meerdere bestanden bestaat. Werk je met Cloud9, dan kun je het beste de code knippen en plakken in een teksteditor en lokaal opslaan (met extensie `.py`) en vervolgens uploaden naar DomJudge.

Nadat je je oplossing hebt ingezonden zal de server deze beoordelen. Dit zal meestal niet meer dan een paar seconden duren. Als de beoordeling klaar is zal je één van de volgende resultaten zien:

- `CORRECT` - je inzending is correct, je bent klaar met de opgave!
- `WRONG-ANSWER` - je programma geeft een verkeerd antwoord. Controleer nog eens of jouw uitvoer exact overeenkomt met de voorbeelduitvoer, en of je niet iets over het hoofd gezien hebt in de opgave.
- `RUN-ERROR` - je programma is gecrasht. Werkt je programma wel goed op de voorbeeldinvoer? Dan is er waarschijnlijk een speciaal geval van invoer waar je geen rekening mee hebt gehouden.
- `TIMELIMIT` - je programma deed er te lang over om antwoord te geven. Waarschijnlijk kom je ergens in een oneindige loop terecht; check je programma nog eens. Een andere mogelijkheid is dat jouw oplossing gewoon niet efficiënt genoeg is.
- `COMPILER-ERROR` - de server kon geen kaas maken van je bestand. Je kunt de details van de foutmelding zien door op de inzending te klikken. Een mogelijkheid is dat je een shebang-statement in je bestand had staan. Dit is om veiligheidsredenen niet toegestaan in DOMJudge.
- `NO-OUPUT` - Je programma genereert geen output, wellicht ben je vergeten een `print` statement toe te voegen.

Je zult waarschijnlijk geen andere foutmeldingen tegenkomen. Zie je die toch, steek dan even je hand op.

Tijdens de toets kun je je vragen stellen aan ons, je hoeft dus *geen* gebruik te maken van de *Clarification Requests*. Het zou wel kunnen dat als er een foutje zit in een vraag, of als we zien dat er iets geks gebeurt bij een van je inzendingen, we dit via de clarifications meedelen. Hou dit dus wel in de gaten!

A Oertellen ★☆☆☆☆

Tijdens dit vak hebben bepaalde studenten al praatjes gegeven over verschillende getallenstelsels. Maar naast de gewone binaire, octale, decimale en vigesimale (20 tallig) stelsels zijn historisch gezien nog veel meer stelsels gebruikt. Denk aan bijvoorbeeld Romijnse cijfers of Mayacijfers. Recent is er echter een nieuw getalstelsel gevonden dat al 50 000 jaar geleden werdt gebruikt. Vanwege de lange bestaansgeschiedenis wordt dit ook wel het oertallig-stelsel genoemd. Wetenschappers begrijpen nog niet precies hoe het werkt, maar hebben wel een steen gevonden waarop enkele conversies tussen decimaal stelsel een oertallig-stelsel worden voorgedaan. Nu is de vraag of studenten van Programmeren in de Wiskunde uit deze gevonden voorbeelden kunnen destilleren hoe het oertallig-stelsel werkt. Schrijf een programma dat decimale getallen kan vertalen naar oertallige getallen en oertallige getallen kan vertalen naar decimale getallen.

A.1 Voorbeeld:

Input	Output
5	Ug ug ug ug ug!

A.2 Voorbeeld:

Input	Output
Ug ug!	2

A.3 Voorbeeld:

Input	Output
Ug!	1

A.4 Voorbeeld:

Input	Output
3	Ug ug ug!

A.5 Voorbeeld:

Input	Output
8	Ug ug ug ug ug ug ug ug!

A.6 Hints:

- Let op leestekens en hoofdletters!
- Er bestaan alleen natuurlijke getallen groter of gelijk aan 1 in het oertallig-stelsel. Uiteraard worden er geen onvertaalbare inputs gegeven.
- Vergeet niet dat je programma beide kanten op moet werken (van en naar oertallig-stelsel)

B Hekjes verven ★☆☆☆☆

Een boer wil alle hekjes in zijn vele weilanden van een nieuwe kleur voorzien. Uit ervaring weet hij dat voor het verven van één hekje 5 liter verf nodig is. Omdat hij te lui was om de hekjes handmatig te tellen, heeft hij met een drone foto's gemaakt van zijn weilanden. Deze foto's heeft hij vervolgens omgezet naar ASCII code, waarbij een hekje uiteraard met het symbool # wordt aangeduid.

Aan jou de taak om een script te schrijven dat voor een gegeven weiland berekent hoeveel verf er nodig is om alle aanwezige hekjes te verven.

Als input ontvang je eerst een geheel getal N dat de lengte van het weiland specificeert. Vervolgens krijg je N regels ASCII code die tezamen het weiland representeren. Zie onderstaande voorbeelden voor de opmaak van je output.

B.1 Voorbeeld:

Input	Output
11 ################################## # ___________ # # < Booeeeeeeee > # # -------------- # # \ ^__^ # # \ (oo)_________ # # (__)\ )\/\ # # ||----w	# # || || # # # ##################################	Om de hekjes in dit weiland te verven heb je 430 liter verf nodig

B.2 Voorbeeld:

Input	Output
3 SJ#skj30# 2##asd21 jko90ml	Om de hekjes in dit weiland te verven heb je 20 liter verf nodig

B.3 Voorbeeld:

Input	Output
4 dit weiland is leeg	Om de hekjes in dit weiland te verven heb je 0 liter verf nodig

C Tetranacci ★★☆☆☆

De Fibonacci getallen zijn recursief gedefinieerd:

$$F(0) = 0,$$
$$F(1) = 1,$$
$$F(n) = F(n-1) + F(n-2)$$

In deze opgave definieëren we de Tetranacci getallen als:

$$G(0) = 0,$$
$$G(1) = 0,$$
$$G(2) = 0,$$
$$G(3) = 1,$$
$$G(n) = G(n-1) + G(n-2) + G(n-3) + G(n-4)$$

Schrijf een programma die voor $0 < n \leq 1000$ het n-de Tetranacci getal berekent.

C.1 Voorbeeld:

Input	Output
5	2

C.2 Voorbeeld:

Input	Output
10	56

C.3 Voorbeeld:

Input	Output
15	1490

C.4 Voorbeeld:

Input	Output
1000	806129655435690133750530176...

Het bovenstaande getal is afgekort, en bestaat normaal uit 284 cijfers. De implementatie die gevraagd wordt zou het volledige getal moeten geven.

D Swipe ★★☆☆☆

Waar je vroeger op je Nokia 3310 met T9 relatief snel berichtjes kon SMS-en naar een persoon, kan je tegenwoordig met een "swipe" bewegingen een tweet naar de hele wereld verzenden! Wat was swipe ook al weer? Stel je wilt het woord "python" intypen; je legt je vinger op op de "p" op je full-HD retina display, en dan beweeg je je vinger over het toetsenbord naar de "y", "t", "h", "o" en tot slot "n", zonder je vinger op te tillen.

Je mobiel krijgt dan dus de tekst "poiuytghjiokn" door. Maar hoe kan je mobiel weten welk woord je bedoeld? Een producent van B-merk smartphones vraagt of jij dit probleem kan oplossen. Je krijgt als input eerst het getal k, het aantal woorden in het woordenboek van de smartphone. Vervolgens komen die k woorden, elk op hun eigen regel, in de input. Daarna komt het getal n, het aantal swipe-inputs. Vervolgens komen er n swipe-inputs, ieder op hun eigen regel. Geef voor elke swipe-input een woord dat matched uit het woordenboek, of een "?" als geen enkel woord uit het woordenboek matched. Een swipe-input matched met een woord als;

- de swipe-input alle letters uit het woord op volgorde bevat
- de eerste letter in de swipe-input gelijk is aan de eerste letter van het woord
- de laatste letter in de swipe-input gelijk is aan de laatste letter van het woord
- let op herhalende letters! Mensen swipen reeksen van dezelfde letter niet expliciet. Zie het voorbeeld, "paard"; de "a" in "poijhgffdsasdfrfd" staat voor de dubbele "a" in "paard"

Gegeven is dat $n, k < 200$. Regels in de input bevatten maximaal 25 tekens.

D.1 Voorbeeld:

Input	Output
5 python pot paard kaas slang 6 poiuytghjiokn poijhgffdsasdfrfd paart paardk pard kaaaaas	 python paard ? ? paard kaas

D.2 Voorbeeld:

Input	Output
3 a aaaaa abab 6 a a aaaaa aaaaa ababa ababab	 a aaaaa a aaaaa aaaaa abab

D.3 Hints:

- Als er meerdere woorden bij een input passen maakt het niet uit welke je als output geeft
- Woorden in het gegeven woordenboek en swipe-inputs bestaan alleen uit kleine letters (dus geen hoofdletters)
- Zowel de swipe-input “a” als “aba” matchen het woord “aaaa”. In het eerste geval heeft de gebruiker de letter “a” niet meerdere malen geswiped, maar wou de gebruiker wel meerdere malen de letter “a” intypen. In het tweede geval is de “b” tussendoor een onbedoelde letter in de swipe-input, en is ten minste een van de letters “a” meervoudig bedoeld.

E Rekenen met de *Reverse Polish Notation* ★★★☆☆

De *Reverse Polish Notation* (RPN) is een manier om ingewikkelde berekening zo weer te geven dat er nooit haakjes zijn. De notatie werkt door eerst de operanden te geven en pas daarna de operatie. De berekening `5-3` wordt in RPN dus geschreven als `5 3 -`. We kunnen uiteraard ook meerdere operaties door elkaar gebruiken. In dat geval is de uitkomst van de ene operatie een operand van de ander, en dat zien we ook weer terug in RPN. Zo kunnen we `(5-3) * 2` weergeven als `5 3 - 2 *`. Een iets uitgebreider voorbeeld is de berekening `5 + (1 + 2) * 4 - 3` die er in RPN als `5 1 2 + 4 * + 3 -` uitziet.

In deze opgave krijg je een berekening in RPN als input, en als output moet je de uitkomst van de berekening geven. De input bestaat altijd uit één regel, waarbij de symbolen gescheiden worden door spaties. Een symbool is altijd een geheel getal of een operator (`+,-,*,/`). De uitvoer moet een regel zijn met daarop één getal: de uitkomst van de berekening, afgerond op 3 decimalen.

E.1 Voorbeeld:

Input	Output
`5 1 2 + 4 * + 3 -`	`14.000`
`5 2 + 3 /`	`2.333`
`12 13 *`	`156.000`
`2 3 -`	`-1.000`
`-2 3 *`	`-6.000`

E.2 Hints:

- Om een getal `x` om te zetten naar een string die het getal met drie decimalen weergeeft, kan je de python-opdracht `"{0:.3f}".format(x)` gebruiken.

F *Reverse Polish Notation* omschrijven ★★★★☆

In deze opgave gebruiken we dezelfde notatie als in de opgave "Rekenen met de *Reverse Polish Notation*".

In deze opgave krijg je een berekening in RPN als input, en als output moet je dezelfde berekening in infix notatie geven. De input bestaat altijd uit één regel, waarbij de symbolen gescheiden worden door spaties. Een symbool is altijd een geheel getal of een operator (`+`,`-`,`*`,`/`). De infix notatie is de 'gewone' notatie waarin we berekeningen meestal opschrijven. Bij de uitvoer is het verder van belang dat om elke stap in de berekening haakjes staan (precies één paar haakjes per operator), en dat aan beide kanten van een operator spaties staan. Laat de getallen in infix notatie in dezelfde volgorde staan als in de Reverse Polish Notation.

F.1 Voorbeeld:

Input	Output
`5 1 2 + 4 * + 3 -`	`((5 + ((1 + 2) * 4) ) - 3)`
`5 2 + 3 /`	`((5 + 2) / 3)`
`12 13 *`	`(12 * 13)`
`-2 3 *`	`(-2 * 3)`

G *Reverse Polish Notation* efficiënt omschrijven ★★★★★

In deze opgave gebruiken we dezelfde notatie als in de opgave "Rekenen met de *Reverse Polish Notation*".

In deze opgave krijg je een berekening in RPN als input, en als output moet je dezelfde berekening in infix notatie geven. De input bestaat altijd uit één regel, waarbij de symbolen gescheiden worden door spaties. Een symbool is altijd een geheel getal of een operator (`+`,`-`,`*`,`/`). De infix notatie is de 'gewone' notatie waarin we berekeningen meestal opschrijven. Het verschil met de vorige opgave is dat je **zo veel mogelijk haakjes moet weglaten**. Verder moeten aan weerszijde van een operator spaties staan.

Je mag bij deze opgave uiteraard uitgaan van de standaard bewerkingsvolgorde en associativiteit. Dus: vermenigvuldiging en deling hebben voorrang op optellen en aftrekken, maar tussen vermenigvuldiging/deling en optellen/aftrekken onderling wordt geen onderscheid gemaakt. Verder zijn optelling en vermenigvuldiging associatief, en aftrekken en delen links associatief.

Let op: Het is dus *niet* de bedoeling dat je operaties gaat veranderen om haakjes te besparen. Ook hoef je niet de structuur van een berekening te veranderen door haakjes te verplaatsen, je hoeft alleen haakjes *weg te laten* (ten op zichte van opgave F).

G.1 Voorbeeld:

Input	Output
`5 1 2 + 4 * + 3 -`	`5 + (1 + 2) * 4 - 3`
`5 1 2 + 4 + + 3 -`	`5 + 1 + 2 + 4 - 3`
`5 1 2 + 4 + - 3 -`	`5 - (1 + 2 + 4) - 3`
`1 2 / 3 4 / 5 6 * /`	`(1 / 2 - 3 / 4) / (5 * 6)`
`12 13 *`	`12 * 13`
`-2 3 *`	`-2 * 3`

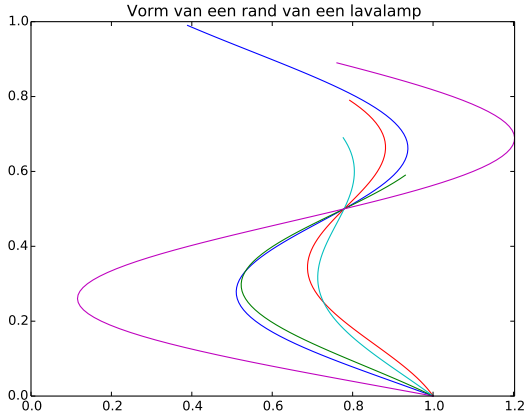
H Lavalamp ★★★★☆

Pieter wil een lavalamp kopen. Er zijn verschillende lavalampen te koop, elk met een unieke vorm.

In het algemeen wordt de vorm van een lavalamp beschreven door de functie:

$$x = (1 - (a + by)\sin(2\pi y))\exp(-y^2)$$

voor bepaalde waarden van a en b op het interval $y \in [0, h]$ rond de y-as te wentelen. Hieronder wordt een deel van de contour van de lavalamp laten zien voor vijf verschillende sets van parameters:



Deze komen overeen met het eerste voorbeeld hieronder. In het bijzonder is Pieter geïnteresseerd in het vinden van de lavalamp met het grootste volume.

Als input krijg je een aantal combinaties van parameters a, b, h. Precies:

- Eerst krijg je een geheel getal $0 < N \leq 10$, het aantal verschillende lampen.
- Dan volgen op N regels drie (floating point) waarden: $0 < a_i, b_i, h_i \leq 1$ gescheiden door spaties.

Als output dien je de index van grootste lavalamp te geven (in termen van volume) als geheel getal tussen 1 en N. De volumes van de lavalampen zullen altijd minimaal op twee decimalen nauwkeurig verschillen.

H.1 Voorbeeld:

Input	Output
5 0.4 0.2 1.0 0.3 0.5 0.6 0.1 0.5 0.8 0.2 0.1 0.7 0.8 0.3 0.9	5

De volumes V_i van de eerste twee lampen uit deze input zijn afgerond ongeveer 1.680 en 0.977.

H.2 Voorbeeld:

Input	Output
2 0.26 0.44 0.60 0.80 0.40 0.69	2