NADA, KTH **Betygsgradering:** D-A
2D1312 2006-2007(python)

170 Aritmatrisen

Läs anvisningarna på kurshemsidan!!!

Varudeklaration: Datastrukturer, slumptal.

En trivsam lekplats för Ture Teknolog och hans glada vänner, som givetvis vill använda sina raster på KTH till att ytterligare förbättra sina matematiska förmågor, är den så kallade Aritmatrisen. Aritmatrisen är en kvadratisk matris med ca 4 rutors sida, som lekledaren Osquar ritat upp med krita på marken. I var och en av matrisens rutor har han skrivit ett någorlunda stokastiskt tal i intervallet 1 - 100. När nu Ture kommer och vill leka, hittar Osquar genast på ett till stokastiskt tal, detta i intervallet 1 - 300, som han säger till Ture. Nu är det Tures uppgift att, påhejad av entusiastiska kamrater, hoppa mellan rutorna i matrisen och med hjälp av +, - och talen han hoppar till försöka bilda det tal Osquar sa.

Du ska skriva ett program som simulerar den här leken på så sätt att användaren får se Aritmatrisen med alla dess siffror på skärmen. Ture står på matrisen, markerad på något lämpligt sätt. OBS! Det är inte lämpligt att täcka siffran med Ture. Båda ska synas samtidigt. Användarens roll blir att skriva instruktioner för hur Ture ska hoppa, och om han ska använda + eller -. En körning kan se ut så här:

15	98	75	82
22	19	8	83
73	91	*35*	99
64	50	37	65

```
u=upp, n=ner, h=höger, v=vänster u2=upp uv=upp-till-vänster osv

Du kan hoppa högst 2 rutor åt gången.
Avsluta genom att hoppa ut ur matrisen!
Om du ledsnat och vill avbryta, ge a!

Du ska försöka få 231.Just nu har du 35.
Vart vill du hoppa? uh
+ eller -? +
```

Här har spelet just börjat, med att Tures startposition slumpats ut i matrisen. * visar Tures plats. Nästa gång matrisen visas, kommer rutan där 83 står att vara markerad, och texten kommer att visa att användaren samlat på sig 118 (= 35 + 83) poäng. Spelet avbryts genom att användaren låter Ture hoppa ur matrisen, eller ger något avbrottstecken (a ovan). Om han hoppar ut och då har lyckats få det givna antalet poäng (här 231), har han klarat uppgiften, och belönas med att få veta det, samt hur många hopp han behövde.

På grund av begränsad benlängd, kan Ture inte hoppa längre än två rutor horisontellt eller vertikalt, eller en ruta diagonalt. Man ska inte kunna hoppa på stället, dvs räkna en ruta två gånger i följd.

Du ska inte lösa uppgiften genom att låta användaren ange koordinater för den ruta han vill hoppa till, utan enligt exemplet genom antal steg åt ett visst håll utgående från den position han för tillfället står på.

Matrisens storlek ska givetvis vara lätt modifierbar.

Extrauppgift, betyg A: Gör ett grafiskt användargränssnitt till ditt program så att användaren kan klicka med musen på den ruta han vill hoppa till.

Extrauppgift, betyg B: Utvidga ditt program så att flera personer kan spela samtidigt! Användaren ska få välja 1 - 4 spelare på plan. Givetvis ska övre gränsen för antal spelare vara lätt att ändra. Alla spelarna slumpas ut på planen och markeras med olika tecken. Sedan turas man om att hoppa till dess någon spelare fått rätt summa och hoppat ut, eller alla spelare hoppat ut.

Reglerna för hoppandet är som ovan, med några tillägg. Två personer kan inte stå på samma ruta. Man kan inte hoppa över en annan person. Om en spelare försöker göra något av dessa fel, ska ett felmeddelande komma upp, och han ska få försöka igen. Givetvis ska inte två personer någonsin starta på samma ruta. Om en av spelarna blir inträngd av de andra, så att han inte kan hoppa, eller hans enda hoppalternativ blir att hoppa av planen, ska han tillåtas att stå över ett hopp. Finns något enda möjligt hopp, måste han dock utnyttja detta. I fallet högst fyra spelare, blir enda inträngningsmöjligheten ett hörn, men ditt program ska ju klara att man ökar gränsen för antal spelare godtyckligt, så du måste ta hand om alla inträngningsfall.

När du nu kan tänkas få fyra spelare på plan, ska matrisen utökas till fem rutors sida, så att mer plats finns.

Tänk på, att om en spelare lämnat planen (utan att få rätt summa), ska han inte markeras på den, och han ska inte ombes att hoppa.

Extrauppgift, betyg B: Se till att problemet alltid är lösbart (d v s att talet Osquar säger alltid kan fås)! Detta kan göras på flera sätt, men ganska enkelt är att utgå från att om man kan få talet 1 på något vis, går problemet att lösa. I så fall är det ju nämligen bara att hoppa till sig en mängd ettor tills man fått den önskade summan.

Hur ser man då till att det alltid går att få 1? Så enkelt som att alltid slumpa fram minst en etta på planen får du inte göra det för dig. Nej, då blir lösningen för lättgenomskådad. Gör i stället så här:

- Skapa planen som förut, med ett tal på varje plats i matrisen.
- Välj slumpvis en ruta i matrisen. Vi kallar denna ruta A.
- Börja på en annan ruta än A och simulera några (ca 5) stokastiska hopp inom matrisen med slumpvis plus och minus. Se till att inte hoppa på A! Nu har du fått summan S.
- Om S = 1, ändra ingenting.
- Om S$\neq$1 men S ligger i samma intervall som talen på planen (dvs 1 - 100), sätt talet i A till S - 1!
- Om S ligger utanför intervallet, fortsätt att hoppa stokastiskt, men nu med plus om S<1 och minus om S>100, tills du hamnar inom intervallet. Gör sedan enligt föregående punkter!

Fundera igenom algoritmen. Du ska ha förstått hur den fungerar vid redovisningen!