TENTAMEN

Programmering Grundkurs (HI1900)

Skrivtid 13:15-18:15

Tisdagen 26 april 2011

Tentamen består av 8 sidor

Hjälpmedel
Förutom dator med installerad *Code::Blocks*,
Utforskaren, *Acrobat reader* och *Notepad* (inga andra program),
den kurslitteratur som använts under kursen, samt egna anteckningar,
programlistningar och böcker. Dock inga egna disketter, CD-ROM eller USB-minne.

- Under `W:\PROV\C` finns program- och datafiler som kan komma till användning vid lösandet av uppgifterna. Kopiera över dessa till ditt konto.
- Till alla uppgifter ska ett program levereras i form av källkod (C eller CPP-fil). Dina bidrag lägger du i en katalog i roten på `H:`. Katalogen ska ha samma namn som prefixet i din *mailadress*. Exempelvis för Kalle Kula: `HDI02KEKA`. Namnen på lösningarna ska ges `UPPG1.C` till `UPPG8.C`. De är endast dessa filer som kommer att bedömas.
- Inled varje program med ditt namn på en kommentarrad.

Rättningen görs genom att programmen körs ett antal gånger för olika indata. Om resultatet överensstämmer med det förväntade bedöms programmet som korrekt och ger 2 poäng. Om ett program ej kan kompileras utan fel, är det knappast troligt att det kommer att ge några poäng. I det fall där programmet läser från och eller skriver till en fil, testas programmet oftast med en annan fil än den bifogade.

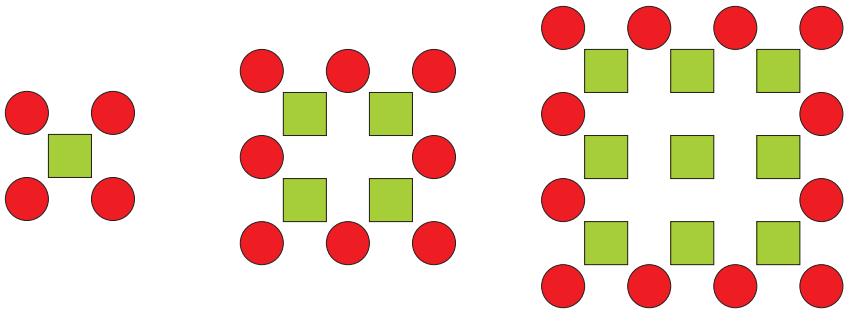
Betygsgränser (HI1900):

Poäng	14-16	12-13	10-11	8-9	7	6
Betyg	A	B	C	D	E	Fx

Resultatet anslås på kursens hemsida `ingforum.haninge.kth.se/c` i kodat skick och i Bilda

Lycka till!

Håkan Strömberg

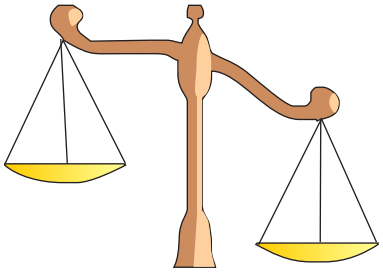


Figur 1:

1 – Trädgården

En trädgårdsarkitekt har designat en modell för fruktträdgårdar, en kvadratisk historia där ett antal äppelträd omgärdar ett antal päronträd. I figur 1 ser vi tre storlekar från denna modell, med $2, 3$ och 4 äppelträd utefter en sida. Skriv ett program som tar emot uppgift om antalet äppelträd utefter en sida och som bestämmer det totala antalet äppel- respektive päronträd i hela trädgården. Ett körningsexempel:

```
Antal äppelträd utefter en sida ? 4
Det behövs 12 äppelträd
Det behövs 9 päronträd
```



Figur 2:

2 – Vägning

Vi har 9 kulor, numrerade 1...9 och en balansvåg. Vi vet att alla kulor utom en väger lika mycket. Den avvikande kulan är lättare.

Med hjälp av följande schema kan man ta reda på vilken av kulorna som är lättare än de övriga.

Vägning 1	Resultat	Vägning 2	Resultat	Kula
1,2,3 ↔ 4,5,6	V	4↔5	V	5
			L	6
			H	4
	L	7↔8	V	8
			L	9
			H	7
	H	1↔2	V	2
			L	3
			H	1

Först placerar vi alltså kulorna 1, 2, 3 i vänster vågskål och 4, 5, 6 i höger. Om resultatet blir att högra (H som i högra) vågskålen sjunker ned (raderna 7 till 9 i tabellen), vet vi att någon av kulorna 1, 2, 3 är den lättare. Vi placerar då 1 i den vänstra och 2 i den högra vågskålen. Om vi får jämvikt (L som i lika) går vi till rad 8 i tabellen och ser att det är kula 3 som är den lättare. På liknande sätt kan man alltid ta reda på vilken kula som är lättare genom endast två vägningar.

Skriv ett program som ställer rätt frågor (två stycken), tar emot ett av svaren V, L eller H och som efter två vägningar (frågor) skriver ut numret på den kula som är lättare. Ett par körningsexempel:

```
1,2,3 <-> 4,5,6 ? L
7 <-> 8         ? V
Kula 8 är lättare

1,2,3 <-> 4,5,6 ? V
4 <-> 5         ? V
Kula 5 är lättare
```

3 – Snigeln i brunnen

En snigel befinner sig på botten av en torrlagt brunn. Snigeln önskar komma upp ur brunnen, som är d meter djup. Under dagen (halva dygnet) kryper han upp u meter med konstant hastighet, men på natten (andra halvan av dygnet), när han sover glider han ned n meter.

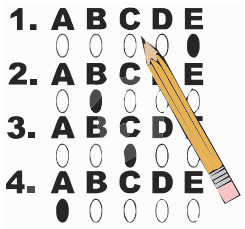
Skriv ett program som tar reda på hur lång tid (i dygn räknat) han behöver för att komma upp ur brunnen. Indata till programmet är brunnens djup (d meter), antal meter uppåt under dagtid (u meter) och antal meter nedåt under natten (n meter). Dessa variabler ska vara flyttal. Snigeln startar sin resa på morgonen och hinner tillryggalägga u meter innan det blir natt (om han nu inte råkar hinna ända upp innan dess). Ett körningsexempel:

```
Brunnens djup   ? 10.0
Uppåt på dagen  ? 6.0
Nedåt på natten ? 3.5

Snigeln når friheten efter 2.42 dygn
```

Första dagen hinner han 6 meter upp. På natten glider han ned 3.5 meter till en nivå på 2.5 meter. Nästa dag hinner han upp till 8.5 meter innan han går till sängs och glider ned till nivån 5 meter under natten. Den tredje dagen kommer han att nå sitt mål och han behöver inte hela halva dygnet till detta och kommer därför upp efter 2.42 dygn (avrundat till två decimaler).

4 – Ny tentamen



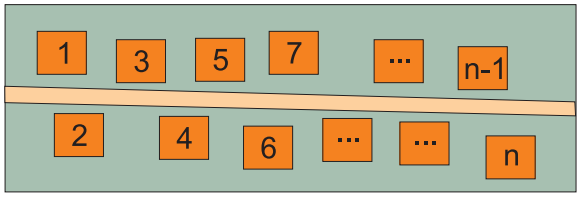
Figur 3: *En tentamen med 4 frågor och 5 svarsalternativ*

Istället för den här ganska besvärliga tentamen, skulle man kunna tänka sig en så kallad *multiple choice*-tentamen, det vill säga en tentamen med $f < 100$ frågor och med $2 \leq a \leq 10$ svarsalternativ per fråga, och där det krävs $g \leq f$ för att bli godkänd.

I valet av f, a, g, vill man ta reda på hur stor chansen är, att den student som inte har läst en enda rad i boken, ska kunna klara sig genom att enbart chansa på svaren.

Skriv ett program som tar reda på hur många procent av 10000 gissande studenter som kommer att bli godkända på en tentamen med f frågor som har a alternativ (endast ett är rätt) och där det gäller att få g rätt för att bli godkänd. Ett körningsexempel:

```
Antal frågor      ? 16
Antal Alternativ  ? 3
Gräns för godkänt ? 8
12.7% av chansande studenter kommer att bli godkända
```



Figur 4:

5 – Det dubblerade gatunumret.

Adam har fått en märklig arbetsuppgift! Han ska klistra nummer på gatans postlådor. Husen är numrerade från 1 till $n < 100$, men ett av husen har två postlådor. De kommer att få samma nummer (förstås) åtskilda av A och B.

För tvåsiffriga nummer klistrar han upp två siffror. Innan han startar arbetet roar han sig med att summera alla siffror han har i påsen och får summan s. Han frågar då Bertil, om han kan säga vilken postlåda, vars nummer som är dubblerat. Bertil påstår att man måste ha en dator för att kunna besvara den frågan, men eftersom han inte kan programmera får du nu uppgiften att skriva programmet. Programmet ska fråga efter vilken summa Adam fick och svara med vilket lådnummer som är dubblerat

Ett körningsexempel:

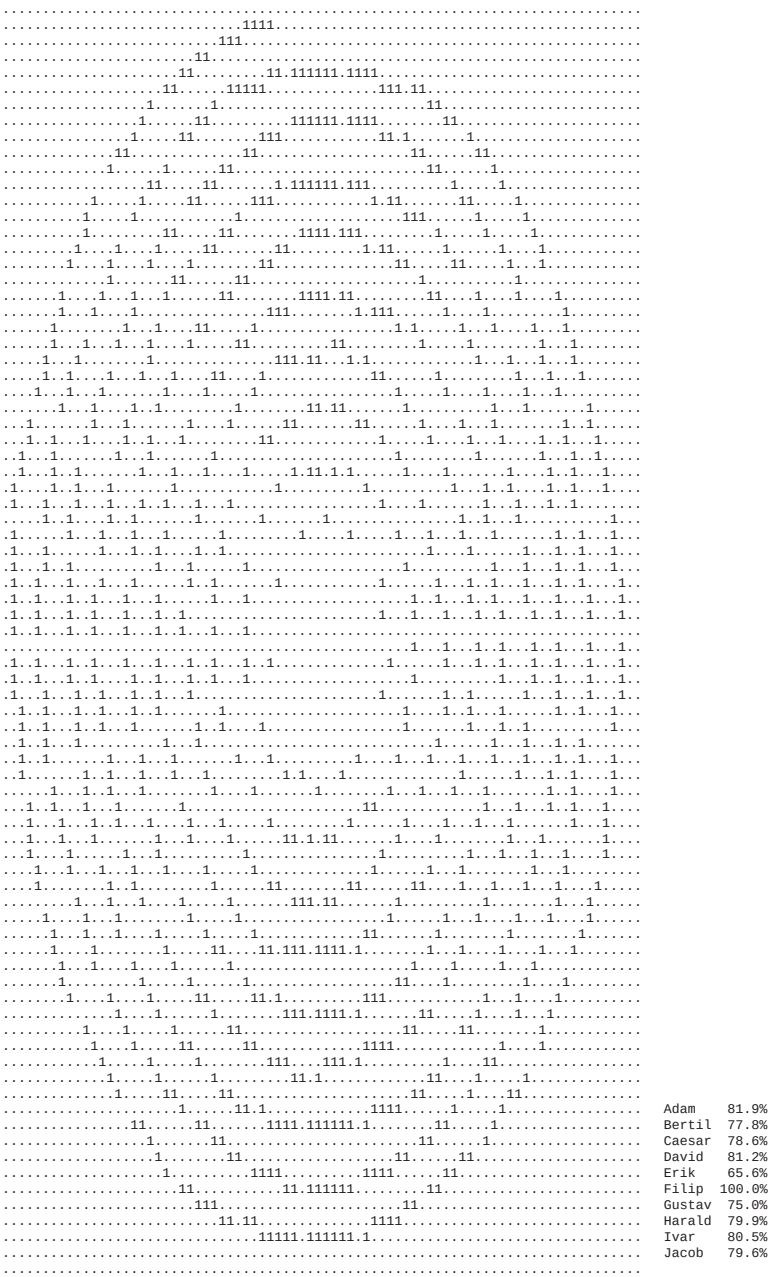
```
Summan av alla siffrorna ? 66
Lådnumret 6 är dubblerat
```

Om ingen lådnummer hade varit dubblerat hade siffersumman blivit

$$1+2+3+4+5+6+7+8+9+1+0+1+1+1+2+1+3+1+4 = 60$$

Om det nu finns 6A och 6B blir summan 66. Hade vi tagit med ett hus till hade siffersumman blivit $60+1+5 = 66$ och inget utrymme finns att dubblera något nummer. Hade vi tagit till ett hus mindre hade summan blivit 55 och vi kan aldrig nå upp till summan 66 genom att dubblera något lådnummer≤ 15.

6 – Fingeravtryck

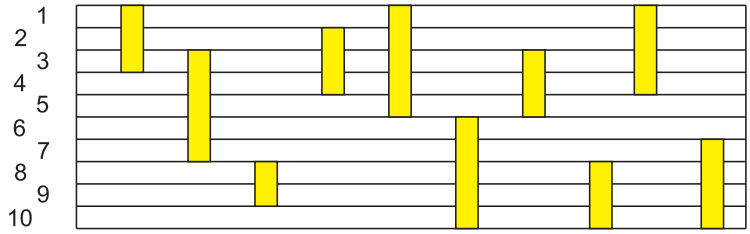


Figur 5: *Fingeravtryck funnet på mordplatsen*

Figur 5 föreställer ett fingeravtryck. På filen `fingeravtryck.txt` finns flera, bland annat ett funnet på mordplatsen. Skriv ett program som matchar det funna fingeravtrycket med ett antal fingeravtryck från misstänkta. Fingeravtrycken på filen består av 80 rader med 80 tecken på varje. Endast tecknen `1` (siffran ett) och `.` (punkt) förekommer. Filen inleds med ett tal $n \leq 20$, som anger hur många fingeravtryck den innehåller. På nästa rad finns texten `Misstänkt`. Därefter följer fingeravtrycket funnet på brottsplatsen. Därefter följer $n-1$ avtryck som inleds med ett namn (≤ 12 tecken) och därefter tillhörande avtryck.

Programmet ska jämföra det första avtrycket med övriga, tecken för tecken och ange i en tabell hur stor procent av fingeravtrycket som överensstämmer med det misstänkta. Ett körningsexempel: Resultatet ser vi (av utrymmesskäl) till höger om avtrycket. Av allt att döma heter mördaren Filip!

7 – Hinderbanan



Figur 6:

I figur 6 ser vi en hinderbana med 10 banor. De gula markeringarna är hinder (alla lika höga). Får en löpare fritt välja, väljer han troligtvis en bana med så få hinder som möjligt.

Skriv ett program som bestämmer vilken av de 10 banorna som har det minsta antalet hinder. Information om banans utseende finns på filen `hinder.txt`. Filen inleds med ett tal $n < 100$ som anger på hur många platser hinder placerats ut. Därefter följer n rader med två tal s_1 och $s_2 \geq s_1$. s_1 anger från och med vilken bana hindret sträcker sig och s_2 anger till och med vilken bana hindret når. Ett körningsexempel:

```
Bana 6 med 2 hinder
```

8 – Personnummer, Namn och Telefonnummer

```
struct person1{
  char personnr[12];
  char telenr[12];
};
```

```
struct person2{
  char personnr[12];
  char namn[48];
};
```

Till denna uppgift hör två binära filer `telefonnr.dat` skapad med `struct person1` och `namn.dat`, skapade med `struct person2`.

Som framgår av postbeskrivningarna ovan innehåller den första filen personnummer och telefonnummer för ett antal personer. Den andra filen ett antal personnummer och namn. Båda filerna innehåller < 10000 poster.

Man önskar nu kombinera dessa filer och vill först ta reda på hur många *fullständiga* uppgifter det finns, det vill säga för hur många personer man känner personnummer, namn och telefonnummer.

```
För 8055 personer finns tre uppgifter
```