2016-11-10.kl.14-19

Tentaupplägg

TIPS 1:

Läs igenom ALLA uppgifterna. Välj den du känner är lättast först. Det kan gärna ta 10-20 minuter. Försök skriva saker som kan vara problem i uppgifterna. Är det något du absolut kommer att fastna på så kanske det är fel uppgift att ge sig på. Tiden du lägger på att noga läsa uppgifterna tjänar du in på att välja rätt uppgift.

TIPS 2:

Kolla ibland till kommunikationsfönstret. Det kan ha kommit information till alla utan att ni skickat in en fråga. Kanske gäller det dig också (d.v.s. den uppgift du jobbar med).

TIPS 3:

Om ni har problem med kompilator, Emacs eller annat som INTE har med uppgifterna att göra, räck upp handen så kommer en assistent. Detsamma gäller om hur man kopierar givna filer
" cp given_files/* . " eller liknande.

Frågor om själva uppgifterna tar vi i första hand via tentasystemet.

I körexemplen har vi markerat det som användaren matar in på tangentbordet med ***fet och kursiverad*** stil. Tänk på att körexemplen bara är *ett* exempel på när programmet körs. Testa ditt program noga och tänka över hur programmet skall fungera vid andra indata.

Vi hinner normalt sett inte svara på frågor de sista 10 utminutera på tentan. Då ägnar vi all tid åt att rätta uppgifter för att alla skall hinna få svar innan ni går hem.

Betygsgränser:	**Tid**	**DI/EL**	**SVP**
1 poäng	19:00	Betyg 3	Betyg G
3 poäng	18:00	Betyg 4	
3 poäng	16:45		Betyg VG
3 poäng	16:30	Betyg 5	
4-6 poäng	18:00	Betyg 5	
4-6 poäng	18:15		Betyg VG

Bonustid från laborationerna tillkommer till dessa tidsgränser. Tiden för betyg 3/G överstiger dock aldrig 5 timmar.

OBS: Delpoäng delas inte ut på uppgifterna. För att få poäng på en uppgift måste man alltså lösa uppgiften helt (och dessutom på ett generellt sätt och enligt specifikation).

Lycka till med tenterandet och hoppas att alla får G på minst en uppgift idag.
M.v.h.
/Torbjörn (examinator)

2016-11-10.kl.14-19

Uppgift 1 - Det snurrande strecket [1p]

Skriv ett program som ritar ut ett streck som snurrar. Ditt program skall alltså göra en *animation*. För att lösa detta behöver ditt program kunna rensa bort all text i terminalen och även kunna göra en kort paus. Här är kommandona för detta:

`delay 0.1;`	Pausar i 0.1 sekunder.
`Put(Ascii.Esc & "[2J" &` `  Ascii.Esc & "[1;1H");`	Rensar all text i terminalen så att den blir blank och flyttar markören till övre vänstra hörnet.

Programmet skall fortsätta att köra animationen tills användaren avslutar med Ctrl-C.

För att få ett snurrande streck så kan man använda sig av tecknen '/' '-' '\' '|'.
För att det skall bli snyggt bör sekvensen vara:

\ | / - \ | / - o.s.v.

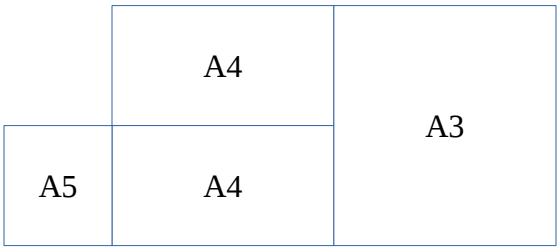
TIPS: Lägg dessa tecken i ett fält.

TIPS 2: Exempel på hur kommandona som nämns ovan används finns i filen *delay_and_clear.adb*.

Körexempel:
Det är svårt att visa hur detta skall se ut eftersom det är en animation. Tänk dig ett roterande streck uppe i terminalens vänstra hörn. Direkt till höger om detta kommer terminalens markör att blinka.

Uppgift 2 - A4-Papper [1p]

Den här tentan är utskriven på A4-papper. Något du kanske inte tänkt på är att ett A3-papper är precis dubbelt så stort som ett A4, och att ett A5 är precis hälften så stort som ett A4!



Således är ett A2 dubbelt så stort som ett A3, och ett A1 är dubbelt så stort som ett A2. Ett A0 är dubbelt så stort som ett A1 och är precis 841 x 1189mm (vilket är nästan exakt 1 m² till ytan). Kort och gott skulle man kunna formulera det hela på ett generellt sätt så här:

Ett AN-papper är:
841,0 x 1189,0 mm, då N = 0.
Annars, storleken hos ett A(N-1)-papper, vikt dubbelt på långsidan.

Skriv ett program där användaren får mata in vilken pappersstorlek som den vill veta hur stor den är. Programmet skall sedan beräkna detta med hjälp av **rekursion**. Resultatet skall skrivas ut som två flyttal (bredd x höjd) med en decimal*.

Körexempel 1:

```
Vilket papper vill du veta storleken på: A4
Dimensionerna på ett A4-papper är 210.3 x 297.3 mm.
```

Körexempel 2:

```
Vilket papper vill du veta storleken på: A0
Dimensionerna på ett A4-papper är 841.0 x 1189.0 mm.
```

Körexempel 3:

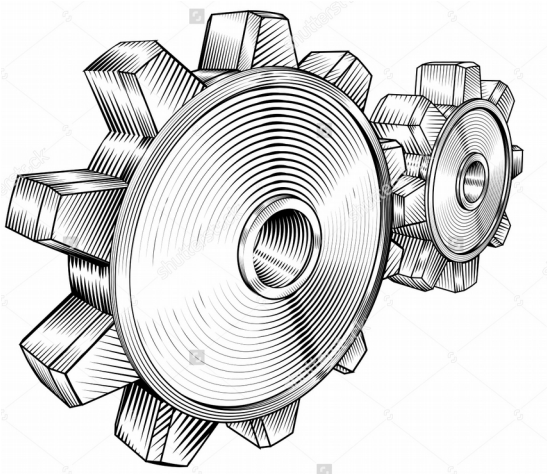
```
Vilket papper vill du veta storleken på: A15
Dimensionerna på ett A4-papper är 4.6 x 6.6 mm.
```

* De officiella pappersstorlekarna är förmodligen avrundade till närmsta millimeter, men vi bortser från detta i denna uppgift.

2016-11-10.kl.14-19

Uppgift 3 - Kugghjul [2p]

Vi tänker oss ett antal kugghjul som sitter ihopkopplade på rad. Kugghjulen är av olika storlek, men dess kuggar är ungefär lika stora. Om man snurrar på det första hjulet ett visst antal varv, hur många varv kommer det sista hjulet snurra? Vi tar ett exempel:



I figuren till vänster har vi två kugghjul på rad. Det vänstra har 10 kuggar och det högra har 8. Om vi snurrar 2 varv "framåt" på det vänstra så kommer det högra snurra "bakåt" 2*10 = 20 kuggar, vilket är 20 / 8 = 2.5 varv. Eftersom det blir motsatt håll så kan vi anse att detta är -2.5 varv.

Skriv ett program där användaren först matar in hur många varv som den kommer snurra på det kugghjul som är längst till vänster. Där efter matar användaren in en sekvens av heltal som motsvar antal kuggar på varje kugghjul i sekvensen. Talen separeras med ett blanksteg, och det är maximalt 50 stycken som kan matas in. När användaren är färdig skriver den en punkt och trycker sedan enter.

Därefter skall programmet mata ut hur många varv som varje kugghjul snurrar (utskrivet med en decimal). Observera att användaren kan mata in ett negativt tal för den första snurrningen, och det behöver inte vara ett heltal. Du kan utgå ifrån att det är minst två kugghjul som matas in.

Körexempel 1:

```
Mata in hur många varv som kugghjulet längst till vänster snurras: 2.0
Mata in sekvensen kugghjul: 10 8.

Kugghjulen snurrar: 2.0 -2.5
```

Körexempel 2:

```
Mata in hur många varv som kugghjulet längst till vänster snurrar: 2.0
Mata in sekvensen kugghjul: 10 8 6 4 20.

Kugghjulen snurrar: 2.0 -2.5 3.3 -5.0 1.0
```

Körexempel 3:

```
Mata in hur många varv som kugghjulet längst till vänster snurrar: -2.5
Mata in sekvensen kugghjul: 8 10.

Kugghjulen snurrar: -2.5 2.0
```

2016-11-10.kl.14-19

Uppgift 4´ - Långa mellanrum [2p]

Skriv ett program som läser igenom sin egen kod. Programmet skall ta reda på vilken som är den längsta sekvensen mellanrum som hänger ihop. Sådana sekvenser får inte hänga ihop över flera rader. Ta följande exempelprogram

```
with Ada.Text_IO; use Ada.Text_IO;

procedure Testa is
begin
  Put("Hejsan");
end Testa;
```

Den längsta sammanhängande sekvensen mellanrum är 2 (de två mellanslagen innan Put).

Körexempel:

```
Den längsta sammanhängande sekvensen mellanrum i koden är 2 tecken
lång.
```