Örebro universitet
Institutionen för teknik
Thomas Padron-McCarthy (Thomas.Padron-McCarthy@tech.oru.se)

Tentamen i
Programmering C

för D1 m fl, även distanskursen

måndag 21 augusti 2006 kl 8:00 - 13:00 i L001

Hjälpmedel:	Inga hjälpmedel.
Poängkrav:	Maximal poäng är 40. För godkänt betyg (3 respektive G) krävs 20 poäng.
Resultat och lösningar:	Meddelas på kursens hemsida senast måndag 28 augusti 2006.
Visning och frågestund:	Måndag 28 augusti 2006 kl 12:00-12:30 i mitt rum (T2220). Efter visningen kan tentorna hämtas på expeditionen.
Examinator och jourhavande:	Thomas Padron-McCarthy, telefon 0707-347013.

- Skriv tydligt och klart. Lösningar som inte går att läsa kan naturligtvis inte ge några poäng. Oklara formuleringar kommer att misstolkas.
- Skriv den personliga tentamenskoden på varje inlämnat blad. Skriv *inte* namn eller personnummer på bladen.
- Skriv bara på en sida av papperet. Använd inte röd skrift.
- Antaganden utöver de som står i uppgifterna måste anges. Gjorda antaganden får inte förändra den givna uppgiften.

LYCKA TILL!

Prioritet och associativitet hos operatorerna i C

De viktigaste operatorerna:

Prioritet	Kategori	Operator	Associativitet
Högsta	Unära postfixoperatorer	(), [], ->, ., ++, --	vänster
	Unära prefixoperatorer	!, ++, +, -, *, &, sizeof, (*typ*)	höger
	Multiplikation mm	*, /, %	vänster
	Addition mm	+, -	vänster
	Jämförelser	<, <=, >=, >	vänster
	Likhetsjämförelser	==, !=	vänster
	Logiskt OCH	&&	vänster
	Logiskt ELLER	||	vänster
Lägsta	Tilldelning	=, +=, -=, *=, /=, %=	höger

Uppgift 1 (1 p)

Skriv formeln nedan som en tilldelningssats i C.

$$x = \sqrt{4ab + \sqrt{3 - r}}$$

Uppgift 2 (1 p)

Vilka värden har följande uttryck?

a) 8 + 3 / 2 + 1

b) 8 % 3 / 2 * 1

Uppgift 3 (1 p)

i är en heltalsvariabel. Ange värdet på i då följande kod körts.

```
i = 8;
while (i > 0) {
  if (i % 2)
    i /= 3;
  else
    i = i - 10;
}
```

Uppgift 4 (1 p)

Vi behöver en posttyp (med C-terminologi en **struct**) som kan användas för att lagra data om ett TV-program. Det som ska finnas med i posten är:

- TV-programmets namn, till exempel **Cops**. Programnamnet kan vara högst 30 tecken långt.
- Namnet på den kanal som programmet visas på, till exempel **Discovery Mix**. Ett kanalnamn kan vara högst 20 tecken långt.
- Tiden när programmet startar. Den anges med timme och minut, uttryckt som två heltal.
- Tiden när programmet slutar. Den anges också med timme och minut, uttryckt som två heltal.

Vi arbetar bara med dagens TV-program, så det behövs inget datumfält.

Definiera posttypen. Den ska heta **struct Program**.

Uppgift 5 (1 p)

Definiera en variabel av typen **struct Program** och initiera den med data om TV-programmet **CSI** som visas kl 21:00 - 22:00 i **Kanal 5**.

Uppgift 6 (2 p)

Skriv en funktion som heter **visa_program**, som skriver ut ett TV-program (som i uppgift 4) på skärmen. Funktionen tar programposten som parameter, och skriver ut den. Exempel på hur urskriften ska se ut:

```
21:00 - 22:00 (Kanal 5): CSI
```

Så här ser funktionshuvudet ut:

```
void visa_program(struct Program programmet)
```

Uppgift 7 (2 p)

Skriv en funktion som heter **krockar**, som tar två TV-programsposter som argument, och som returnerar ett sant värde om de två TV-programmen krockar, dvs om deras sändningstider överlappar. Annars ska funktionen returnera värdet 0.

Det räcker att de krockar alls, så två TV-program som sänds 21:00 - 21:30 och 21:29 - 21:40 krockar, men två TV-program som sänds 21:00 - 21:30 och 21:30 - 21:40 krockar inte.

Så här ser funktionshuvudet ut:

```
int krockar(struct Program program1, struct Program program2)
```

Uppgift 8 (2 p)

Man kan bara sända ett program åt gången i en TV-kanal, så om vi hittar två TV-program som krockar fast de sänds i samma kanal, så måste det vara fel nånstans.

Skriv en funktion som heter **fel**, som jämför två TV-program, och som returnerar ett sant värde om de två TV-programmen både krockar och sänds i samma kanal. Annars ska funktionen returnera värdet 0. Anropa funktionen **krockar** från uppgift 7 för att avgöra om programmen krockar.

Så här ser funktionshuvudet ut:

```
int fel(struct Program program1, struct Program program2)
```

I den här och alla andra uppgifter på tentan gäller: Om du ska anropa en funktion från en tidigare uppgift, behöver du inte skriva koden för den funktionen på nytt. Du får också anropa funktionen även om du inte gjort uppgiften där man skulle skriva den.

Uppgift 9 (2 p)

Skriv en funktion som heter **las_program**, och som läser in data om ett TV-program. Funktionen ska skriva ut lämpliga ledtexter på standardutmatningen, och läsa in data från standardinmatningen (som normalt är kopplad till tangentbordet). Du får själv välja om du vill att funktionshuvudet ska se ut så här:

```
struct Program las_program()
```

eller så här:

```
void las_program(struct Program* p)
```

Uppgift 10 (2 p)

Faktaruta:
Man ska se upp när man låter ett C-program hantera svenska personnummer, för ett tiosiffrigt personnummer som **6312111658** går inte in i ett 32-bitars heltal, vilket är vad många datorer jobbar med. Men i den här uppgiften antar vi att vi har en 64-bitarsdator, och att vanliga heltal av typen **int** lagras med 64 bitar. Då går det bra att hantera så stora tal.

Skriv en funktion som tar ett personnummer, lagrat som ett heltal, som argument, och som skriver ut det med ett streck i mitten, som man brukar skriva ut personnummer.

Om vi skickar heltalet **6312111658** till funktionen, ska det alltså skrivas ut som **631211-1658**.

Uppgift 11 (5 p)

Skriv ett C-program som slumpar fram ett personnummer, och sen skriver ut det. Personnumret ska se rimligt ut, så vi kräver att månadsnumret ska vara mellan 1 och 12, och att dagnumret ska vara mellan 1 och 31. Vi bryr oss inte om att en del månader har mindre än 31 dagar, och kontrollsiffran på slutet behöver inte vara riktig.

Exempel:

- **631211-1658** är ett rimligt (och dessutom riktigt) personnummer.
- **631211-1659** räknar vi som ett rimligt personnummer, även om kontrollsiffran är fel.
- **631411-1658** räknar vi inte som ett rimligt personnummer, för det finns ingen månad nummer 14.

Om du vill kan du använda funktionen från uppgift 10 för att skriva ut personnumret, men det är inget krav.

Tips: Funktionen **rand()** ger ett slumpmässigt heltal, som är större än eller lika med 0, och som är mindre än någon ganska stor övre gräns.

Uppgift 12 (5 p)

Skriv ett C-program som upprepat beräknar bensinkostnader.

Det går till så att programmet frågar efter bensinförbrukningen för en bil (mätt i liter per mil), bensinpris (mätt i kronor per liter), och sen efter längden på de sträckor man kört (mätt i kilometer). Man matar in de sträckor som man kört, och avslutar med att mata in en sträcka med längden 0. Då skrivs den sammanlagda bensinkostnaden ut.

Detta upprepas tills man matar in bensinförbrukningen 0.

(Vi antar alltså att bensinpriset inte ändras under perioden.)

Ett exempel (med användarens inmatning i *kursiv stil*):

```
Ange bensinförbrukningen för din bil (liter per mil, avsluta med 0): 1.0
Ange bensinpriset (kronor per liter): 10.00
Ange längden på en sträcka (kilometer, avsluta med 0): 100
Ange längden på en sträcka (kilometer, avsluta med 0): 100
Ange längden på en sträcka (kilometer, avsluta med 0): 0
Sammanlagd bensinkostnad: 200.00
Ange bensinförbrukningen för din bil (liter per mil, avsluta med 0): 0.67
Ange bensinpriset (kronor per liter): 12.90
Ange längden på en sträcka (kilometer, avsluta med 0): 14
Ange längden på en sträcka (kilometer, avsluta med 0): 24.4
Ange längden på en sträcka (kilometer, avsluta med 0): 13.1
Ange längden på en sträcka (kilometer, avsluta med 0): 0
Sammanlagd bensinkostnad: 44.51
Ange bensinförbrukningen för din bil (liter per mil, avsluta med 0): 0
```

Uppgift 13 (5 p)

Skriv ett C-program som läser in två TV-program, med hjälp av funktionen **las_program**.

C-programmet ska sedan anropa funktionen **fel** för att kontrollera om TV-programmen sänds samtidigt i samma kanal. I så fall ska ett felmeddelande skrivas ut, och C-programmet ska avslutas.

Om TV-programmen inte sänds samtidigt i samma kanal, ska funktionen **krockar** anropas för att avgöra om programmen krockar, och programmet ska skriva ut ifall det var en krock eller inte.

Uppgift 14 (5 p)

Skriv ett C-program som läser in TV-program från standardinmatningen, med hjälp av funktionen **las_program**, och sparar dem på en textfil som heter **program.txt**. Programmet ska avslutas när man matar in ett TV-program med sändningstiden 0:00 - 0:00.

Välj själv vilket format textfilen ska ha, och beskriv hur det formatet ser ut.

Glöm inte att beskriva formatet som textfilen ska ha.

Uppgift 15 (5 p)

På en binärfil som heter **program.bin** finns ett antal program (dock högst 1000 stycken) lagrade i binärform, enligt posttypen **struct Program**. Skriv ett C-program som läser in TV-programmen från filen, och berättar om några av programmen krockar. Använd funktionen **krockar**.

C-programmet behöver inte tala om vilka TV-program som krockar, eller hur många,

utan bara *att* det finns minst en krock.