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

Tentamen i

Programmering grundkurs och Programmering C

för D1 m fl, även distanskursen

onsdag 14 januari 2009 kl 14:00 - 19:00

Gäller som tentamen för:
DT1016 Datateknik A, Programmering grundkurs, provkod 0100
DT1007 Datateknik A, Tillämpad datavetenskap, provkod 0410
DT1006 Datateknik A, Programmering C, distans, provkod 0100

Hjälpmedel:	Inga hjälpmedel.
Poängkrav:	Maximal poäng är 40. För godkänt betyg (3 respektive G) krävs 23 poäng. För den som följt campuskursen hösten 2008 ger varje i tid inlämnad inlämningsuppgift med deadline en extra poäng. Den som *inte* gått campuskursen hösten 2008 får dessa (tre) extrapoäng ändå.
Resultat och lösningar:	Meddelas via e-post eller på kursens hemsida, http://www.aass.oru.se/~tpy/c/2008-2009-p2/, senast onsdag 4 februari 2009.
Återlämning av tentor:	Efter att resultatet meddelats kan tentorna hämtas på institutionen. Man kan också få sin rättade tenta hemskickad.
Examinator och jourhavande:	Thomas Padron-McCarthy, telefon 070-73 47 013.

- Skriv tydligt och klart. Lösningar som inte går att läsa kan naturligtvis inte ge några poäng. Oklara och tvetydiga 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.
- Skriv gärna förklaringar om hur du tänkt. Även ett svar som är fel kan ge poäng, om det finns med en förklaring som visar att huvudtankarna var rätt.

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)

Vilka värden har följande uttryck?

a) 4 * 3 - 3 * 2

b) 4 - 3 / 3 * 2

c) 4 / 3 - (3 % 2)

Uppgift 2 (1 p)

a, **b** och **c** är heltalsvariabler. Ange värdet på a, b och c när följande kod har körts.

```
a = 1; b = 2; c = 3;
if (a < b + c)
    c = c + 2;
while (a < b + 1) {
    ++c;
    ++a;
}
```

Uppgift 3 (1 p)

x och **y** är variabler av typen **float**. **p1** och **p2** är variabler av typen **pekare till float**. Ange värdet på x och y när följande kod har körts.

```
p1 = &x;
*p1 = 3.14;
p2 = &y;
*p2 = *p1 + 1;
p2 = p1;
```

Uppgift 4 (1 p)

Skriv en funktion som heter **medel**, som tar tre flyttal av typen **double** som argument, och returnerar talens medelvärde. (Ledtråd: Man får medelvärdet genom att summera de tre talen, och dela summan med tre.)

Uppgift 5 (2 p)

Skriv en funktion som heter **arraymedel**, som tar en array med flyttal av typen **double** som argument, tillsammans med ett heltalsargument som anger antalet tal i arrayen, och returnerar talens medelvärde.

Scenario till uppgift 6-14

Sveriges meteorologiska och hydrologiska institut, SMHI, är en statlig myndighet som bland annat har till uppgift att ta fram väderprognoser.

För att kunna framställa sina väderprognoser behöver de göra temperaturmätningar på olika ställen. Hittills har mätningarna gjorts på fasta stationer runt om i Sverige, men för att spara pengar ska SMHI nu i stället ha en enda, mobil, mätstation, som åker runt i Sverige och mäter temperaturen.

SMHI:s nya mobila väderstation.

Uppgift 6 (1 p)

Temperaturmätningarna ska lagras i poster ("structar") av typen **struct Temperatur**. En temperaturpost innehåller själva **temperaturen** (angiven i Celsius), som är ett flyttal. Dessutom finns ett klockslag, till exempel **11:55** och vilken plats, till exempel **Örebro**, som mätningen gjordes på. Platssnamn kan vara högst tio tecken långa (som **Lindesberg**).

Definiera posttypen **struct Temperatur**.

Uppgift 7 (1 p)

Definiera en variabel av typen **struct Temperatur** och initiera den med en mätning i **Örebro** kl **12:01** där temperaturen var **-14.0 grader**.

Uppgift 8 (2 p)

Vi vill kunna visa temperaturposternas innehåll på skärmen. Skriv en funktion som heter **visa_temperatur**, som skriver ut en temperaturmätning på skärmen. Funktionen ska ta temperaturposten som parameter. Exempel på hur en utskrift skulle kunna se ut:

```
Örebro kl 12:01: -14.0
```

Uppgift 9 (2 p)

Skriv en funktion som heter **las_temperatur**, och som läser in data om en temperatur. 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 Temperatur las_temperatur()
```

eller så här:

```
void las_temperatur(struct Temperatur *p)
```

Uppgift 10 (3 p)

Skriv en main-funktion som har två lokala variabler av typen **struct Temperatur**, och som läser in data om två temperaturer till dessa variabler med hjälp av funktionen **las_temperatur**. Avslutningsvis ska den använda funktionen **visa_temperatur** för att skriva ut data om den *högsta* av de båda temperaturerna.

I den här och alla andra uppgifter på tentan gäller: Om du ska använda något från en tidigare uppgift eller deluppgift, till exempel anropa en funktion som skrevs i den tidigare uppgiften, så behöver du inte skriva samma kod igen. Du får också göra uppgiften även om du inte gjort den tidigare uppgiften.

Uppgift 11 (1 p)

Variablerna **t1** och **t2** är av typen **struct Temperatur**. Skriv de programrader som behövs för att byta plats på innehållet i de två variablerna.

Uppgift 12 (4 p)

Skriv ett C-program som upprepat läser in temperaturer, med hjälp av funktionen **las_temperatur**. Programmet ska läsa temperaturer ända tills användaren avslutar genom att mata in en temperatur som är under noll grader.

Till slut ska programmet skriva ut medelvärdet av de inmatade temperaturerna. (Den avslutande negativa temperaturen ska inte räknas med.)

Uppgift 13 (7 p)

Skriv ett C-program som låter användaren mata in temperaturmätningar, och sparar dem i en array. Programmet ska läsa temperaturer ända tills användaren avslutar genom att mata in en temperatur med värdet **-100**. Det ska använda funktionen **las_temperatur**. Det ska få plats **1000** temperaturmätningar, och programmet ska avslutas med ett felmeddelande om man försöker mata in fler.

När inmatningen är klar ska programmet spara alla temperaturmätningarna på en fil. Välj själv om det ska vara en text- eller en binärfil.

Uppgift 14 (5 p)

Skriv ett C-program som först frågar efter på en plats, och sen läser filen från uppgiften ovan, och skriver ut både den *högsta* och den *lägsta* temperaturen som uppmätts på den platsen. Programmet ska använda funktionen **visa_temperatur**.

Uppgift 15 (8 p)

a) (4p)

Skriv ett program som använder funktionen **gets** för att läsa in en sträng, som innehåller ett antal ord som åtskiljs med mellanslag. Strängen kan maximalt innehålla 80 tecken. Som utmatning ska programmet tala om hur många ord det var i strängen.

b) (1p)

Man bör egentligen aldrig använda **gets**. Förklara varför!

c) (1p)

Visa (med programkod) hur man kan göra på ett bättre sätt.

d) (2p)

Skriv en funktion som tar en sträng och ett tecken som argument, och som returnerar antalet förekomster i strängen av det tecknet.
