Mikael Rännar
Datavetenskap
Umeå Universitet

Tentamen

Systemnära Programmering, 5p
28 oktober 1994

- Skrivtid 16–22
- Hjälpmedel: **En** av följande böcker:
  - K.A. Jamsa: "*Programmera i C*"
  - A. Kelley & I. Pohl: "*A Book on C*"
  - R. Lafore: "*Microsoft C, Programming for the PC*"
- Maxpoäng 44 (för godkänt krävs 22p, väl godkänt 33p)
- Uppgifterna är slumpmässigt ordnade.
- Börja varje uppgift på *nytt* blad och skriv uppgiftsnummer längst ut till höger på bladet.
- Skriv namn på *varje* blad.
- Tentamensgenomgång (+ återlämning) sker onsdagen den 9 november kl 10:15 i MC413.
- Resultat kommer upp på anslagstavlan så snart jag rättat klart.
  (Kommer även att finnas tillgängligt på WWW.)
- Kursutvärderingen lämnas in tillsammans med tentan eller på tentamensgenomgången.
- Fråga om någonting är oklart!

  Lycka Till!

Uppgift 1 (5 p)

Vad skrivs ut av programmet?

```
#include <stdio.h>

main()
{
    static char s[] = "Att\0vara\0eller\0inte\0vara";
    char *p = s+20;
    char *r[] = {s+13, s+9, s+5, s+1, s};
    char **q = r - 1;

    (*r[3])--;
    printf("%c", *r[0]);
    putchar(*(p+3));
    puts(*(r + 3));
    q[2][5] = 30 + 2;
    puts(*(r+1));
}
```

Uppgift 2 (2 + 3 + 1 p)

Skriv följande macron så säkra som möjligt. Inget argument till macrona ska förändras.

(a) `BETWEEN(a, b, c)`, returnerar sant om $b \leq a \leq c$, annars falskt.

(b) `LSAME(c1, c2)`, ska returnera sant om de två bokstäverna *c1* och *c2* är lika, oavsett om de är stora eller små. Om de är olika ska falskt returneras. (T ex ska `LSAME('a', 'A')` returnera sant, men ej `LSAME('a', 'b')`.) Du får förutsätta att *c1* och *c2* antingen är stora eller små bokstäver.

(c) Förutsatt att man använder heltalsuttryck, kan användandet av macrona ovan resultera i oväntande/oönskade resultat? Motivera ditt svar.

Uppgift 3 (6 p)

Skriv (i ANSI C) en funktion
`int NoWord(char *s1, char *s2)`
som returnerar antalet ord i strängen *s1*, där strängen *s2* innehåller ordseparatorer.
Till exempel: anropet
`NoWord("hej hej i natten", "it")`
returnerar 3 (följande ord räknas: `"hej hej "`, `" na"`, `"en"`). (Får naturligtvis delas upp i flera funktioner om det underlättar. Studera gärna bokens stränghanteringsfunktioner.)

Uppgift 4 (9 p)

Skriv ett program (i ANSI C)

`change [-x -y] [file]`

som byter ut alla förekomster av tecknet x mot tecknet y. Om *file* finns givet ska programmet läsa från den filen annars ska *standard input* användas. Programmet ska alltid skriva på *standard output*. Om tecknet y utelämnas (t ex med anropet "`change -R - infil`", OBS att det andra minustecknet måste skrivas) ska alla förekomster av teckent R tas bort.

Programmet ska kontrollera att anropet är korrekt. (t ex att om `-x` är med så måste `-y` vara med, fast själva y kan vara "tom")

Uppgift 5 (2 + 4 p)

(a) Vad gör följande C-shell script? Vad skrivs ut?

```
#!/bin/csh -f

if ($#argv == 0) then
    echo "Usage: $0 arguments"
    exit 1
endif

set y = 1

foreach x ($argv)
    echo ${y}\: $x
    @ y ++
end

exit 0
```

(b) Konstruera ett rekursivt C-shell script `map` som tar namnet på ett kommando och ett directory som argument. Scriptet ska sedan utföra det specificerade kommandot på **alla** filer i directoriet (utom directoryn och filer som börjar med "."). För alla underdirectoryn i ett directory ska scriptet anropa sig självt rekursivt för att utföra kommandot även på filerna i underdirectoryt. Scriptet ska ha följande syntax:

`map command directory`

Scriptet ska utföra nödvändig kontroll av sina argument samt skriva ut lämpliga felmeddelanden. Ni behöver dock inte kontrollera att kommandot finns.

Tips:

Använd `if (-d` *filename*) för att ta reda på om ett filnamn är ett directory.

Uppgift 6 (4 p)

Förklara noggrant vad följande program gör och skriver ut.

```
#include <stdio.h>

void try_me(void);

main()
{
    printf("\nWhat is your favorite line? ");
    try_me();
    printf("\n\n");
}

void try_me(void)
{
    char c;
    if ((c = getchar()) != '\n')
        try_me();
    putchar(c);
}
```

Uppgift 7 (3 + 3 p)

(a) Vilka 3 huvudtillstånd kan man urskilja vid processhanteringen. Berätta också vad processerna har för sig i de olika tillstånden.

(b) Rita en figur med tillstånden och övergångarna (riktade pilar) mellan de olika tillstånden. Tala om för varje övergång vad som kan få en process att byta just mellan de två tillstånden.

Uppgift 8 (2 p)

Beskriv kortfattat vad som händer när kommandot

```
% grep ENV file.c
```

som användaren matar in tas om hand av C-shell. Dvs inte vad kommandot gör, utan mer vad som händer i processväg.