ÖREBRO UNIVERSITET

INSTITUTIONEN FÖR TEKNIK

Lösningarna till tentamensuppgifterna sätts ut på kurssidan på nätet under eftermiddagen.

Omtentamen i Programmeringsmetodik, 5p (1ED030), 2007-01-12.
[D1, E1, A2, Pr1, SDU2, TDVB]

Hjälpmedel : Inga
Tid : 08:00-13:00
Ansvarig lärare : Christer Lindkvist 303393, 070-3273393

Svar till samtliga uppgifter 1-15 ska skrivas på utdelat extra papper. Använd ett papper till uppgifterna 1-5, två papper till uppgifterna 6-10 och ett papper per uppgift till uppgifterna 11-15 Skriv din tentamenskod på varje inlämnat extra papper.

Den maximala poängen för respektive uppgift står angiven i högermarginalen. Totalt kan 40 poäng erhållas. För betyget 3 krävs ca 20, för betyget 4 ca 28 och för betyget 5 ca 34 poäng.

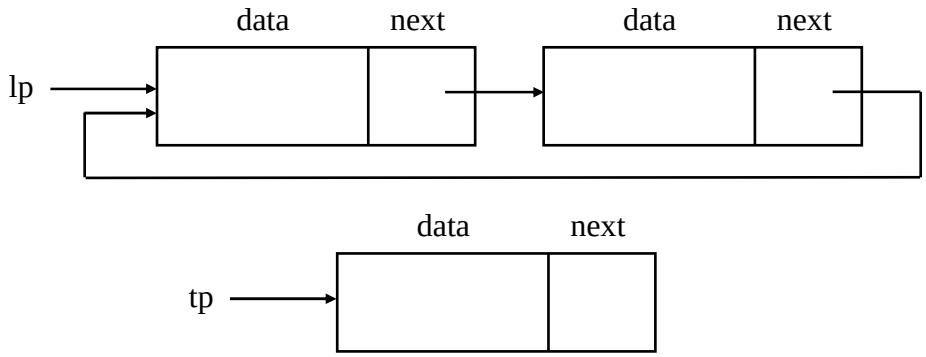
Om inget speciellt anges gäller frågorna Visual C++.

Detta häfte ska du behålla.

Lycka till!

1) Antag att du har en pekare **ip** som pekar på ett minnesutrymme som innehåller ett heltal. Skriv en sats som ökar heltalet med ett. **(1p)**

2) Stega fram pekaren **lp** så att den pekar på nästa länk i den länkade strukturen nedan. Stoppa därefter in länken som **tp** pekar på efter länken som **lp** pekar på. Inga extra pekare får definieras. Den sista structens nextpekare ska alltid peka på den första länken. **(1p)**



3) Antag att du har en 8 bitars unsigned char definierad enligt: **(1p)**

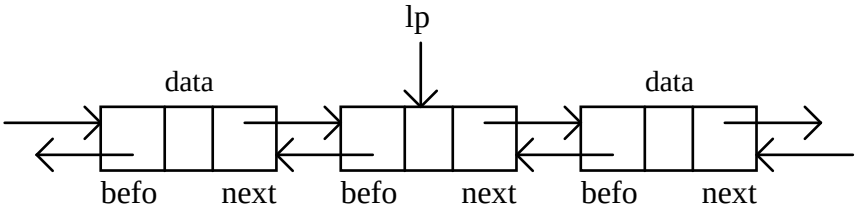
```
unsigned char uch = 0x5a;
```

Ange värdet decimalt av **uch** efter satsen:

```
uch = uch | (uch << 4);
```

4) Ange hur du, med hjälp av funktionerna i Twolist.h (se uppgift 15), tar bort och avallokerar länken efter (till höger om) lp i nedanstående tvåvägslista.

(1p)



5) Visa hur du skriver ut den mittersta strängen i följande vektor av strängar: **(1p)**

```
char *str[] = {"Hello", "ugly" ,"world"};
```

---

6) En växelkassa innehållande mynt kan avbildas som en abstrakt datatyp enligt:

```
/* Specifikation av växelkassa -- kassa.h */

#ifndef KASSA_H
#define KASSA_H

typedef struct
{
   int femtio;   /* Antal femtioöringar */
   int en;       /* Antal enkronor */
   int fem;      /* Antal femkronor */
   int tio;      /* Antal tiokronor */
} kassa;

void skapa_kassa(kassa *kp, int tio, int fem, int en, int femtio);
/* Skapar en kassa med växelmynt eligt parameterlistan */

void las_kassa(char *ledtext, kassa *kp);
/* Skriver ut eventuell ledtext och läser därefter */
/* in en kassa på formen tio, fem, en, femtio      */

void skriv_kassa(kassa k);
/* Skriver ut kassa på skärmen på formen tio, fem, en, femtio */

double varde_kassa(kassa k);
/* Returnerar k:s värde i kr */

int mindre_kassa(kassa k1, kassa k2);
/* Sant om k1:s värde mindre än k2:s, annars falskt */

kassa summa_kassa(kassa k1, kassa k2);
/* Summakassan av k1 och k2 */

#endif
```

Implementera funktionerna **las_kassa** och **skriv_kassa** enl. ovan. **(2p)**

---

7) Implementera funktionerna **skapa_kassa** och **mindre_kassa** i uppgift 6 ovan. Använd gärna **värde_kassa** i implementationen av **mindre_kassa** om du vill! **(2p)**

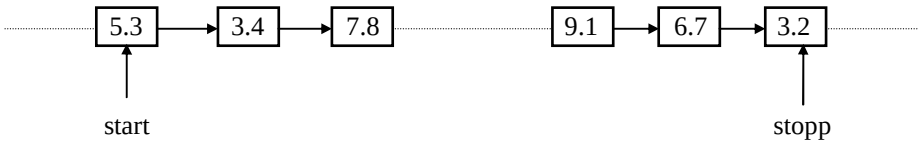
---

8) Fullborda funktionen **positiv**, som ska returnera sant om tal är positivt annars falskt. Inga jämförelseoperatorer finns tillgängliga för tal, så du måste kontrollera om den mest signifikanta biten är nollställd, då är talet positivt. Du kan anta att **int** är 32 bitar. Funktionshuvud enligt: **(2p)**

```
int positiv(int tal);
```

---

9) Antag att du har en envägslista, där länkarna är av samma typ som i uppgift 2 ovan med data i form av reella tal, enligt:



Fullborda funktionen **andra_start** nedan, så att den flyttar fram start **n** st länkar. Passeras stopp under denna förflyttning ska start stanna och peka på samma länk som stopp. Funktionen anropas exempelvis enligt: **(2p)**

```
void andra_start(&start, stopp, 10);
```

```
void andra_start(linktyp **startpek,
                 linktyp *stopp, int n);
```

---

10) Skriv en rekursiv funktion som räknar antalet länkar i en envägslista av liknande typ som i uppgift 2 ovan. Denna lista är dock inte cirkulär. Sista länkens next-pekare är NULL-ställd och pekar inte tillbaka på första länken som den gör i uppgift 2 ovan! Funktionshuvud enligt: **(2p)**

```
int nr_links(linktyp *lp);
```

---

11) Implementera funktionerna **summa_kassa** och **varde_kassa**, enligt uppgift 6 ovan och skriv ett huvudprogram som läser in två kassor, skriver ut summan av dessa och värdet av summakassan. **(5p)**

---

12) Skriv ett fullständigt program som börjar med att fråga efter antalet värden, som en heltalsvektor ska ha och som dynamiskt skapar en sådan vektor. Programmet ska sedan slumpa tresiffriga positiva heltal till vektorn, skriva ut det mittersta värdet i vektorn, om antalet är udda, annars de två mittersta. Avslutningsvis ska det dynamiskt allokerade minnet frigöras. Två körexempel där du matar in det understrukna. **(5p)**

```
Ge antal: 9
213 355 657 651 119 496 879 368 788
Mittersta är 119

Ge antal: 10
755 980 543 534 140 252 315 575 645 143
Två mittersta är 140 och 252
```

---

13) Binärfilen **Kassor.dat** innehåller ett antal växelkassor enligt uppgift 6 ovan. Skriv ett program som läser alla kassor från binärfilen och placerar dessa i en kö. Programmet frågar sedan efter kassavärde (nyckel), tömmer kön och skriver ut alla kassavärden som är **större** än nyckeln. För hantering av kön ska du använda **fifo** nedan. **(5p)**

```c
/* Specifikation av FIFO-lista -- fifo.h */

#ifndef FIFOH
#define FIFOH

#include "kassa.h"
typedef kassa datatyp;     /* Exempelvis */

typedef struct link {
  datatyp data;
  struct link *next;
} linktyp;

void infifo(linktyp **lpp, datatyp d);
/* Stoppar in d i FIFO-lista */

datatyp utfifo(linktyp **lpp);
/* Tar bort data från FIFO-listan */

#endif
```

---

14) Binärfilen **Bilar.txt** innehåller ett antal bilar av den abstrakta datatypen **bil** nedan. Skriv ett program som läser alla bilar från binärfilen och placerar dessa i en hashtabell med öppen adressering. Därefter ska en söknyckel i form av en bil läsas in. Efter sökning i hashtabellen ska sökresultatet skrivas ut. För att hantera hashtabellen skall du endast använda den abstrakta datatypen **openhash** nedan och för att hantera bilar ska du endast använda den abstrakta datatypen **bil**. **(5p)**

```c
/* Specifikation av bilar – bil.h */

#ifndef BIL_H
#define BIL_H

typedef struct bilpost {
  char regnr[7];
  char agare[40];
} biltyp;

void skapa_bil(biltyp *bp, char *regnr, char *agare);
/* Skapar en bil utifrån medskickad data (se las_bil) */

int las_bil(biltyp *bp);
/* Läser in data för en bil från tangentbordet.           */
/* Returnerar 0 då endast RETURN ges som regnr, annars 1  */
/* Om endast RETURN anges som ägare blir ägaren "UNKNOWN" */

void skriv_bil(biltyp b);
/* Skriver ut regnr och agare på skärmen */

int hashfunk_bil(biltyp b);
/* Returnerar summan av ASCII-koderna för b.regnr */

int samma_bil(biltyp b1, biltyp b2);
/* Returnerar 1 om b1 och b2 har samma regnr, annars 0 */

#endif
```

```
/* Specifikation av öppen hashtabell -- openhash.h */

#ifndef OPENHASH_H
#define OPENHASH_H

#include "bil.h"
typedef biltyp datatyp;    /* Exempelvis */

#define HASHTABSIZE 97


void init_openhash(datatyp *htab[]);
/* NULL-ställer hashtabellen */

void into_openhash(datatyp *htab[], datatyp d,
                   int (*hashfunk)(datatyp));
/* Stoppar in d i hashtabell enligt hashfunk */


datatyp *search_openhash(datatyp *htab[], datatyp nyckel,
                         int (*hashfunk)(datatyp),
                         int (*is_equal)(datatyp, datatyp));
/* Söker efter nyckel i hashtabell */

#endif
```

---

15) I textfilen **Kassor.txt** finns ett antal kassor av den abstrakta datatypen kassa, enligt uppgift 11 ovan, på formen tio, fem, en, femtio radvis enligt: **(5p)**

```
5, 12, 6, 4
3, 14, 8, 2
2, 5, 2, 3
8, 1, 5, 3
3, 5, 2, 1
```

Skriv ett fullständigt program som läser alla kassor från filen och stoppar in dessa i en tvåvägslista sorterad enligt mindre_kassa-funktionen. Avslutningsvis ska programmet skriva ut de sorterade kassorna på skärmen med tillägg av värdet enligt:

```
2, 5, 2, 3 = 48.50
3, 5, 2, 1 = 57.50
8, 1, 5, 3 = 91.50
3, 14, 8, 2 = 109.00
5, 12, 6, 4 = 118.00
```

För hantering av kassor ska du endast använda den abstrakta datatypen **kassa** i uppgift 6 ovan och för hantering av tvåvägslistan ska du endast använda den abstrakta datatypen **twolist** nedan.

```
/* Specifikation av tvåvägslista -- twolist.h */

#ifndef TWOLISTH
#define TWOLISTH

#include "kassa.h"
typedef kassa datatyp;

typedef struct twolink {
  enum {head, link} kind;
  struct twolink *befo, *next;
  datatyp data;
} headtyp, linktyp;

void newhead(headtyp **hpp);
/* Skapar en ny tom lista */

void newlink(linktyp **lpp);
/* Skapar en ny tom länk */

void putlink(datatyp d, linktyp *lp);
/* Sätter in data i en länk */

datatyp getlink(linktyp *lp);
/* Returnerar data från länk */

void inlast(linktyp *lp, headtyp *hp);
/* Sätter in länken sist i listan */

void infirst(linktyp *lp, headtyp *hp);
/* Sätter in länken först i listan */

void inpred(linktyp *lp, linktyp *ep);
/* Sätter in första länken före den andra */

void insucc(linktyp *lp, linktyp *ep);
/* Sätter in första länken efter den andra */

void insort(linktyp *lp, headtyp *hp,
            int (*is_less)(datatyp d1, datatyp d2));
/* Sätter in länken sorterad enligt is_less */

linktyp *firstlink(headtyp *hp);
/* Returnerar pekare till första länken i listan */

linktyp *lastlink(headtyp *hp);
/* Returnerar pekare till sista länken i listan */

linktyp *predlink(linktyp *lp);
/* Returnerar pekare till länken före */

linktyp *succlink(linktyp *lp);
/* Returnerar pekare till länken efter */

int is_link(linktyp *lp);
/* Returnerar 1 om länk annars 0 */

int empty(headtyp *hp);
/* Returnerar 1 om listan tom annars 0 */

int nrlinks(headtyp *hp);
/* Returnerar antalet länkar i listan */
```

```
void outlist(linktyp *lp);
/* Tar bort länken från listan */

void elimlink(linktyp **lpp);
/* Tar bort, avallokerar och NULL-ställer länken */

void clearhead(headtyp *hp);
/* Tar bort alla länkar fr†n listan */

void elimhead(headtyp **hpp);
/* Eliminerar och NULL-ställer listan */

#endif
```

Lösningar till tentamen i Programmeringsmetodik, 5p 2007-01-12

1)
```
*ip = *ip + 1;   Exempelvis *ip += 1; och (*ip)++; fungerar också.
                 Det gör däremot inte *ip++; (pekaren uppräknas).
```

2)
```
lp = lp->next;
tp->next = lp->next;
lp->next = tp;
```

3)
```
  01011010
& 10100000
  11111010 = 0xFA = 15*16 + 10 = 250 (0xFA = 0xFF – 0x05 = 255 – 5)
```

4)
```
lp = succlink(lp);
elimlink(&lp);
```

5)
```
printf(str[1]);   Alternativ:  printf("%s", str[1]);
                               puts(str[1]);  // Byter dock rad!
```

6)
```
#include <stdio.h>
#include "kassa.h"

void las_kassa(char *ledtext, kassa *kp)
{
  if (ledtext) printf(ledtext);
  scanf("%d, %d, %d, %d", &kp->tio, &kp->fem, &kp->en, &kp->femtio);
}

void skriv_kassa(kassa k)
{
  printf("%d, %d, %d, %d", k.tio, k.fem, k.en, k.femtio);
}
```

7)
```
void skapa_kassa(kassa *kp, int tio, int fem, int en, int femtio)
{
  kp->tio = tio;
  kp->fem = fem;
  kp->en = en;
  kp->femtio = femtio;
}

int mindre_kassa(kassa k1, kassa k2)
{
  return varde_kassa(k1) < varde_kassa(k2);
}
```

8)
```
int positiv(int tal)
{
  return ((tal & (1 << 31)) == 0);
}
```

9)
```
void andra_start(linktyp **startpek, linktyp *stopp, int n)
{
  int i = 1;

  while (*startpek != stopp && i <= n) {
    *startpek = (*startpek)->next;
    i++;
  }
}
```

10)
```c
int nr_links(linktyp *lp)
{
  if (lp == NULL)
    return 0;
  else
    return 1 + nr_links(lp->next);
}
```

11)
```c
kassa summa_kassa(kassa k1, kassa k2)
{
  kassa sum;

  sum.femtio = k1.femtio + k2.femtio;
  sum.en = k1.en + k2.en;
  sum.fem = k1.fem + k2.fem;
  sum.tio = k1.tio + k2.tio;
  return sum;
}

double varde_kassa(kassa k)
{
  return k.femtio*0.5 + k.en + k.fem*5 + k.tio*10;
}

#include <stdio.h>
#include "kassa.h"

void main()
{
  kassa k1, k2, ksum;

  las_kassa("Ge myntantal i kassa 1 (tio, fem, en, femtio): ", &k1);
  las_kassa("Ge myntantal i kassa 2 (tio, fem, en, femtio): ", &k2);
  ksum = summa_kassa(k1, k2);
  skriv_kassa(ksum);
  printf("\n\nVärde summakassa : %.2f\n", varde_kassa(ksum));
}
```

12)
```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

void main()
{
  int *vp, nr, i;

  printf("Ge antal: ");
  scanf("%d", &nr);
  vp = calloc(nr, sizeof(int));
  srand((unsigned) time(NULL));

  for (i = 0; i < nr; i++) {
    vp[i] = rand() % 900 + 100;
    printf("%d ", vp[i]);
  }

  if (nr % 2)
    printf("\nDet mittersta talet är %d\n", vp[nr/2]);
  else
    printf("\nDe två mittersta talen är %d och %d\n",
           vp[nr/2-1], vp[nr/2]);
  free(vp);
}
```

13)

```c
#include <stdio.h>
#include "kassa.h"
#include "fifo.h"

void main()
{
  linktyp *lp = NULL;
  FILE *bsin;
  kassa k;
  double nyckel, kd;
  int i = 0;

  bsin = fopen("Kassor.dat", "rb");
  fread(&k, sizeof(kassa), 1, bsin);
  while(!feof(bsin)) {
    infifo(&lp, k);
    fread(&k, sizeof(kassa), 1, bsin);
  }
  fclose(bsin);

  printf("Ge nyckelvärde: ");
  scanf("%lf", &nyckel);
  while (lp != NULL) {
    i++;
    k = utfifo(&lp);
    kd = varde_kassa(k);
    if (kd > nyckel) {
      printf("Kassa %d: %.2f > %.2f\n", i, kd, nyckel);
    }
  }
}
```

14)

```c
#include <stdio.h>
#include "bil.h"
#include "openhash.h"

void main()
{
  FILE  *bsin;
  biltyp bil, nyckel, *bp, *hashtab[HASHTABSIZE];

  /* NULL-ställ hashtab */
  init_openhash(hashtab);

  /* Stoppa in alla bilar från filen in i hashtab */
  bsin = fopen("Bilar.dat", "rb");
  fread(&bil, sizeof(biltyp), 1, bsin);
  while(!feof(bsin)) {
    into_openhash(hashtab, bil, hashfunk_bil);
    fread(&bil, sizeof(biltyp), 1, bsin);
  }
  fclose(bsin);

  /* Läs in sökt registreringsnummer */
  printf("Ge sökt regnr, ägaren kan lämnas blank!\n");
  las_bil(&nyckel);

  /* Sök i hashtab och skriv ut resultat */
  if (bp = search_openhash(hashtab, nyckel, hashfunk_bil, samma_bil))
    skriv_bil(*bp);
  else
    printf("Bilen finns ej i registret!\n");
}
```

15)

```c
#include <stdio.h>
#include "kassa.h"
#include "twolist.h"

void main()
{
  FILE *tsin;
  headtyp *hp;
  linktyp *lp;
  int tio, fem, en, femtio;
  kassa k;

  newhead(&hp);
  tsin = fopen("Kassor.txt", "r");
  while (fscanf(tsin, "%d, %d, %d, %d",
                &tio, &fem, &en, &femtio)!= EOF)
  {
    skapa_kassa(&k, tio, fem, en, femtio);
    newlink(&lp);
    putlink(k, lp);
    insort(lp, hp, mindre_kassa);
  }
  fclose(tsin);

  lp = firstlink(hp);
  while (lp != NULL) {
    k = getlink(lp);
    skriv_kassa(k);
    printf(" = %.2f\n", varde_kassa(k));
    lp = succlink(lp);
  }
}
```