ÖREBRO UNIVERSITET

INSTITUTIONEN FÖR TEKNIK

Lösningarna till tentamensuppgifterna sätts ut på kurssidan på nätet idag kl 13.

Denna tenta kommer att vara färdigrättad On 27/8 och kan då hämtas på mitt tjänsterum T2221 mellan 13 och 15. Vid detta tillfälle har du också möjlighet att diskutera rättningen. Tentor som ej hämtats då placeras på studentexpedition.

Omtentamen i Programmeringsmetodik, 5p, Au2, D1, E1, Pr1, 030822.

Hjälpmedel : Inga
Tid : 08-13
Ansvarig lärare : Gunnar Joki 303317, 274825(hem)

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 efter uppgiftens nummer. 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 Borland C.

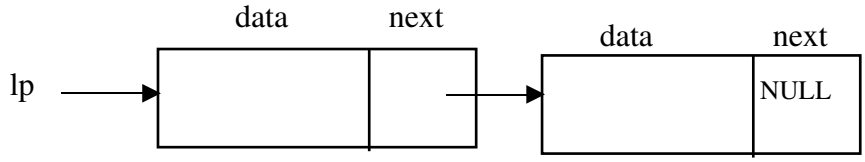
Detta häfte ska du behålla.

Lycka till!

1) (1p) Definiera en reell variabel och ge den ett värde. Definiera sedan en pekare som du sätter att peka på den reella variabeln. Använd sedan pekaren till att skriva ut värdet av den reella variabeln.

---

2)(1p) Skapa dynamiskt en ny länk av linktyp och stoppa in den sist (längst till höger) i den länkade strukturen nedan. Den nya länkens next-pekare ska sättas till NULL.



---

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

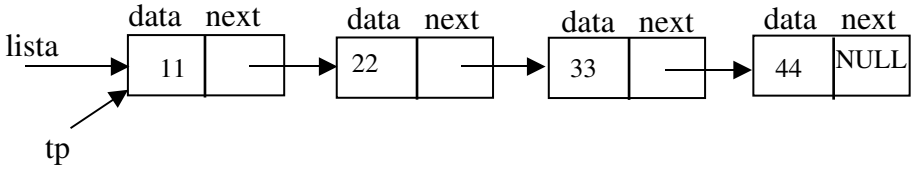
```
unsigned char uch = 0xf;
```

Ange värdet av uch efter satsen:

```
uch &= 3;
```

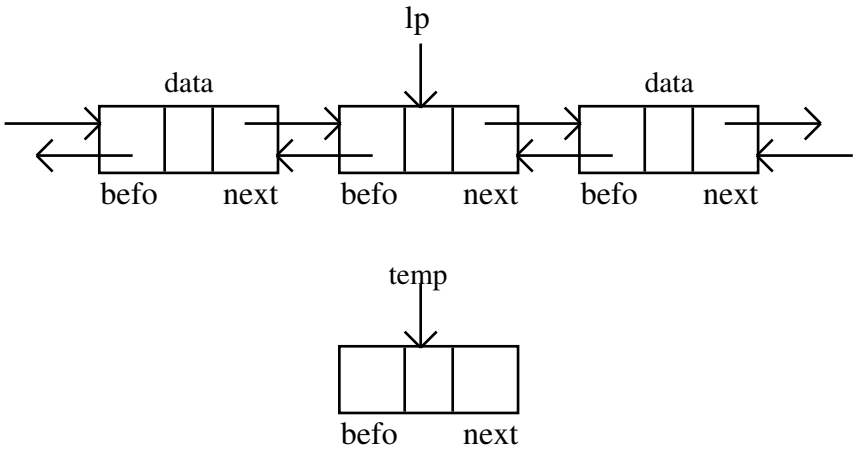
---

4) (1p) Skriv de satser som tar bort och avallokerar den sista (längst till höger) länken i en lista av nedanstående typ. Länken med data 33 ska ha next lika med NULL.



---

5)(1p) Ange hur du ställer om pekarna nedan så att länken temp instoppas efter (till höger om) länken lp i nedanstående tvåvägslista. Inga extra pekare får definieras och listan ska hänga ihop efteråt.



6)(2p) En likströmstvåpol kan avbildas som en abstrakt datatyp med tomgångsspänning (ut) och inre resistans (ri) enligt:

```
/* Tvapol.h */

typedef
struct
{
   float ut;   /* Tomgångsspänning i volt */
   float ri;   /* Inre resistans i ohm */
} tvapol;

void las_tvapol(tvapol *tvp);
/* Läser in ut och ri för tvåpolen */

void skriv_tvapol(tvapol tv);
/* Skriver ut tvåpolens tomgångsspänning och inre resistans */

float kortslut_tvapol(tvapol tv);
/* Returnerar kortslutningsströmmen som ut/ri */

float belasta_tvapol(tvapol tv, float rb);
/* Returnerar belastningsströmmen som ut/(ri+rb) där rb är
belastningsresistansen */

float maxuteffekt_tvapol(tvapol tv);
/* Returnerar största möjliga uteffekt från tvåpolen som fås vid
anpassning då ri är lika med rb och uteffekten ut*ut/(4*ri) */

int mindre_tvapol(tvapol tv1, tvapol tv2);
/* Returnerar sant(1) om den maximala uteffekten från tv1 är mindre
än den från tv2, annars falskt(0) */
```

Implementera funktionen las_tvapol.

---

7)(2p) Implementera funktionen skriv_tvapol ovan.

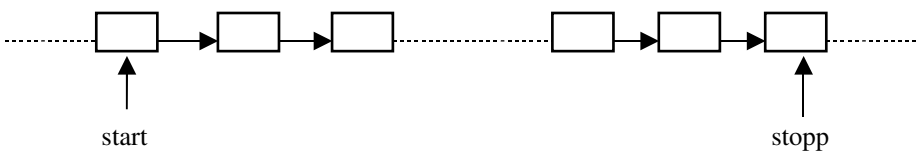
---

8)(2p) Skriv funktionen ettor så att den returnerar antalet ettställda bitar i heltalet tal som består av 32 bitar. Funktionsprototyp enligt:

```
int ettor(int tal);
```

---

9)(2p) Antag att du har en envägslista , där länkarna är av samma typ, linktyp, som i uppgift 2 ovan med data i form av namn med max 20 tecken, enligt:



Skriv funktionen search, så att den söker efter namn (key) och returnerar sant (1) om namnet finns i listan mellan start och stopp, annars falskt (0). Funktionsprototyp enligt:

```
int search(linktyp *start, linktyp *stopp, char *key);
```

10)(2p)Skriv en rekursiv funktion som åstadkommer samma sak som funktionen search i uppgift 9 ovan. Funktionshuvud enligt:

```
int rek_search(linktyp *start, linktyp *stopp, char *key);
```

---

11)(5p)Skriv ett fullständigt program som upprepat (avslut 0) frågar efter hur långt namn som ska läsas in, allokerar dynamiskt minne för namnet, så att det precis ryms och läser in namnet. Efter inläsning ska namnet vändas baklänges genom anrop av den givna funktionen bak_str och skrivas ut. Den givna funktionens prototyp enligt nedan.

```
void bak_str(char *str);
/* Vänder str bak och fram*/
```

---

12)(5p)Fullborda huvudprogrammet nedan så att den stoppar in heltalen från den initierade vektorn i ett binärt träd och sedan frågar efter vilket tal som söks och skriver ut om talet finns i tabellen eller ej. Du ska anropa de givna funktionerna into_bintree och is_in_bintree med prototyper enligt nedan.

```
void into_bintree(nodtyp **npp, int tal);
/* Stoppar in tal i trädet */

int is_in_bintree(nodtyp *np, int key);
/* Returnerar sant(1) om key finns i trädet annars falskt(0)*/

void main()
{
   int vek[5] = {657, 123, 786, 877, 697};
   nodtyp *tree;
   int i, finns, nyckel;

   /* NULL-ställ trädet */

   /* Stoppa in alla tal från vek in i trädet */

   /* Läs in sökt nyckel */

   /* Sök i trädet */

   /* Skriv sökresultat */

}
```

---

13)(5p)Implementera funktionerna kortslut_tvapol, belasta_tvapol och maxuteffekt_tvapol, enligt uppgift 6 ovan och skriv ett huvudprogram som läser in en tvåpol och skriver ut kortslutningsströmmen och den maximala effekten som kan tas ut från tvåpolen.

Programmet ska sedan fråga efter en belastningsresistans som ska anslutas till tvåpolen och skriva ut den ström som tas ut från tvåpolen vid denna belastning.

14)(5p)Skriv ett fullständigt program som läser alla personer med namn och lön i hela kronor från textfilen Loner.txt enligt:

```
B.B   21400
G.G   23500
. . . . . .
```

och lägger dessa på en stack efter det att alla löner räknats upp med en inläst procentuell löneförhöjning, samma för alla. Programmet ska sedan tömma stacken och skriva ut personerna med namn och den nya lönen. För hantering av stacken ska du använda:

```
/* Specifikation av LIFO-lista -- lifo.h */
typedef struct person
{
   char namn[20];
   int lon;
} datatyp;

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

void push(linktyp **lpp, datatyp d);
/* Stoppar in d i LIFO-listan */

datatyp pop(linktyp **lpp);
/* Tar bort data från LIFO-listan */
```

---

15)(5p)Skriv ett fullständigt program som läser alla tvåpoler, enligt uppgift 6 ovan från binärfilen Tvapol.dat till en tvåvägslista sorterad efter tvåpolernas maximala uteffekter enligt mindre_tvapol, som du också ska implementera. Programmet ska sedan fortsätta med att fråga efter önskad effekt för en tvåpol, leta efter den tvåpol i listan som har närmast högre maximal uteffekt och skriva ut denna tvåpol på skärmen. Saknas lämplig tvåpol skrivs detta ut i klartext. För hantering av tvåvägslistan ska du använda:

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

#include ”Tvapol.h”
typedef tvapol 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 från listan */
```

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

Lösningar till tentamen i Programmeringsmetodik, 5p, 030822

1)
```
float x = 2.3;
float *xp = &x;
printf(”%f”, *xp);
```

2)
```
lp->next->next = malloc(sizeof(linktyp));
lp->next->next->next = NULL;
```

3)
```
3
```

4)
```
while(tp->data != 33)
{
   tp = tp_>next;
{
free(tp->next);
tp->next = NULL;
```

5)
```
temp->befo = lp;
temp->next = lp->next;
lp->next->befo = temp;
lp->next = temp;
```

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

void las_tvapol(tvapol *tvp);
{
   printf("Ge tomgångsspänning i volt : ");
   scanf("%f", &tvp->ut);
   printf("Ge inre resistans i ohm: ");
   scanf("%f", &tvp->ri);
}
```

7)
```
#include "Tvapol.h"
#include <stdio.h>


void skriv_tvapol(tvapol tv);
{
   printf(”Tomgångsspänning = %f\n”, tv.ut);
   printf(”Inre resistans = %f\n”, tv.ri);

}
```

8)
```
int ettor(int tal)
{
    int summa = 0, i;

   for (i = 0; i <= 31; i++)
   {
      if ((tal & (1 << i)) != 0)
      {
         summa++;
      }
    }
    return summa;
}
```

9)
```
int search(linktyp *start, linktyp *stopp, char *key)
{
    while (start != stopp)
   {
      if (stricmp(start->data, key) == 0)
      {
         return 1;
      }
      start = start->next;
   }
   return stricmp(stopp->data, key);
}
```

10)
```
int rek_search(linktyp *start, linktyp *stopp, char *key)
{
   if (start == stopp)
   {
      return stricmp(stopp->data, key) == 0;
   }
   else if (stricmp(start->data, key) == 0)
   {
      return 1;
   }
   return rek_search(start->next, stopp, key);
}
```

11)
```
#include <stdio.h>
#include <stdlib.h>

void main()
{
   char *namn;
   int i, len;

   printf("Ge längden av strängen (avslut 0) : ");
   scanf("%d", &len);
   while (len  > 0)
   {
      namn = calloc(len + 1, sizeof(char));
      getchar();
      printf("Ge sträng : ");
      gets(namn);
      bak(namn);
      puts(namn);
      free(namn);
      printf("Ge längden av strängen (avslut 0) : ");
      scanf("%d", &len);
   }
}
```

12)

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

void into_bintree(nodtyp **npp, int tal);
/* Stoppar in tal i trädet */

int is_in_bintree(nodtyp *np, int key);
/* Returnerar sant(1) om key finns i trädet annars falskt(0)*/

void main()
{
   int vek[5] = {657, 123, 786, 877, 697};
   nodtyp *tree;
   int i, finns, nyckel;

   /* NULL-ställ trädet */
   tree = NULL;

   /* Stoppa in alla tal från vek in i trädet */
   for (i = 0; i < 5; i++)
   {
      into_bintree(&tree, vek[i]);
   }

   /* Läs in sökt nyckel */
   printf("Ge sökt nyckel : ");
   scanf("%d", &nyckel);

   /* Sök i trädet */
   finns = is_in_bintree(tree, nyckel);

   /* Skriv sökresultat */
   if (finns)
   {
      printf("Nyckeln finns!");
   }
   else
   {
      printf("Nyckeln finns ej!");
   }
   getch();
}
```

13)

```
#include <stdio.h>
#include <conio.h>
#include "Tvapol.h"

float kortslut_tvapol(tvapol tv)
{
   return tv.ut / tv.ri;
}

float belasta_tvapol(tvapol tv, float rb)
{
```

```
   return tv.ut / (tv.ri + rb);
}

float maxuteffekt_tvapol(tvapol tv)
{
   return tv.ut * tv.ut / (4 * tv.ri);
}

void main()
{
   tvapol t;

   las_tvapol(&t);
   printf(”Kortslutningsström = %f\n”, kortslut_tvapol(t));
   printf(”Maxuteffekt = %f\n”, maxuteffekt_tvapol(t));
   printf(”Ge belastningsresistans : ”);
   scanf(”%f”, &r);
   printf(”Belastningsström = %f\n”, belasta_tvapol(t));
   getch();
}
```

14)

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include "lifo.h"

void main()
{
   linktyp *lp = NULL;
   FILE *tsin;
   datatyp pers;
   float proc;

   tsin = fopen("Loner.txt", "rt");
   printf("Ge löneförhöjning i procent : ");
   scanf("%f", &proc);

   while (fscanf(tsin, "%s %d", pers.namn, &pers.lon) != EOF)
   {
      pers.lon = pers.lon*(1 + proc/100) + 0.5;
      push(&lp, pers);
   }
   fclose(tsin);

   while (lp != NULL)
   {
      pers = pop(&lp);
      printf("%-10s%d\n",  pers.namn, pers.lon);
   }
   getch();
}
```

15)

```
#include <stdio.h>
#include <conio.h>
#include "Twolist.h"

int mindre_tvapol(tvapol tv1, tvapol tv2)
{
   return maxuteffekt_tvapol(tv1) < maxuteffekt_tvapol(tv2);
}

void main()
{
   FILE *bsin;
   headtyp *hp;
   linktyp *lp;
   tvapol t;
   float effekt;
   int found = 0;

   newhead(&hp);
   bsin = fopen("Tvapol.dat", "rb");
   fread(&t, sizeof(tvapol), 1, bsin);
   while(!feof(bsin))
   {
      newlink(&lp);
      putlink(t, lp);
      insort(lp, hp, mindre_tvapol);
      fread(&t, sizeof(tvapol), 1, bsin);
   }
   fclose(bsin);

   printf("Vilken effekt vill du minst ha ? ");
   scanf("%f", &effekt);
   lp = firstlink(hp);
   while (lp != NULL && !found)
   {
      t = getlink(lp);
      if (effekt <= maxuteffekt_tvapol(t))
         found = 1;
      else
         lp = succlink(lp);
   }
   if (found)
   {
      printf("Lämplig tvåpol : \n");
      skriv_tvapol(t);
   }
   else
      printf("Hittar ingen lämplig tvåpol!");
   elimhead(&hp);
   getch();
}
```