ÖREBRO UNIVERSITET

INSTITUTIONEN FÖR TEKNIK

Lösningarna till tentamensuppgifterna sätts ut på kurssidan på nätet i dag kl 19.

Denna tenta kommer att vara färdigrättad Fr 21/1 -05 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, 050114.

Hjälpmedel : Inga
Tid : 14-19
Ansvarig lärare : Gunnar Joki 303317, 274825(hem), 0705474825(mob)

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!

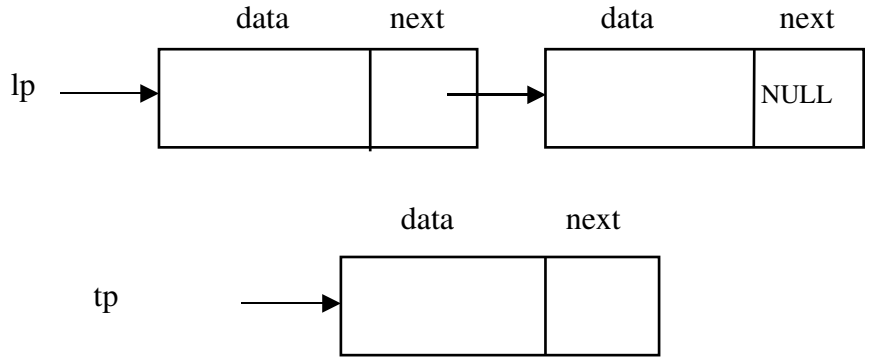
1) (1p) Anta att du har en pekare given enligt:

```
char *cp;
```

Visa hur du använder denna pekare till att dynamiskt allokera minne för ditt namn och sätta in ditt namn i detta minnesutrymme.

---

2)(1p) Stoppa in länken tp först i den länkade strukturen nedan. Inga extra pekare får definieras och lp ska peka på den just insatta första länken.



---

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

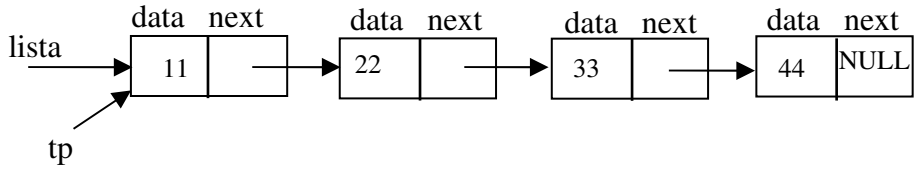
```
unsigned char uch = 0x0f;
```

Ange värdet decimalt av uch efter satsen:

```
uch &= 20;
```

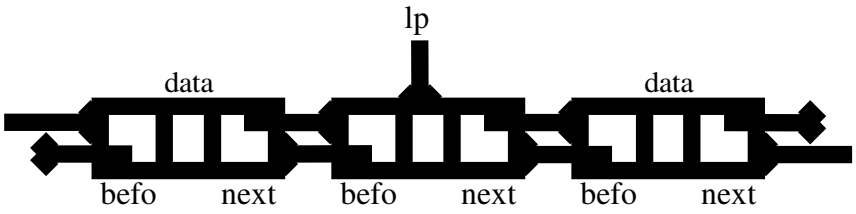
---

4) (1p) Skriv de satser som tar bort och avallokerar den första länken (längst till vänster) nedan. Efter bortplockandet ska lista och tp peka på den andra länken hela listan ska hänga ihop.



---

5)(1p) Ange hur du genom att ställa om pekarna nedan plockar bort länken lp nedan så att listan hänger ihop efteråt.



6)(2p) En fastighet kan avbildas som en abstrakt datatyp med fastighetsbeteckning, taxeringsvärde och försäljningspris enligt:

```
/* Fastighet.h */

typedef
struct
{
   char bet[20];  /* Fastighetsbeteckning ex Lugnet 5*/
   int tax;       /* Taxeringsvärde i hela kr */
   int pris;      /* Försäljningspris i hela kr*/
} fastighet;

void las_fastighet(fastighet *fp);
/* Läser in data med ledtexter */

void skriv_fastighet(fastighet f);
/* Skriver ut data med ledtexter */

int jfr_fastigheter(fastighet f1, fastighet f2);
/* Returnerar sant (1) om f1 har lägre försäljninspris än f2 */

float tax_i_procent_av_pris(fastighet f);
/* Returnerar hur många procent tax utgör av pris */
```

Implementera funktionen las_fastighet.

---

7)(2p) Implementera funktionen skriv_fastighet, i uppgift 6 ovan.

---

8)(2p) Skriv ett huvudprogram som skapar två fastigheter, enligt uppgift 6 ovan, läser in värden till dessa och skriver ut dem i ordning med den fastighet som har det lägsta försäljningspriset först. Använd funktionerna för fastighet i uppgift 6 ovan.

---

9)(2p) Skriv en funktion, udda_paritet, som tar ett tecken som parametrar och som returnerar sant om antalet ettor i tecknets binära ASCII-kod är udda annars falskt.
Funktionsprototyp enligt:

```
int udda_paritet(unsigned char uch );
```

---

10)(2p)Antag att du har en given lista enligt uppgift 4 ovan med data i form av fastigheter enligt uppgift 6 ovan. Skriv ett program som skriver ut medelvärdet av förhållandet mellan taxeringsvärdet och försäljningspriset för fastigheterna i listan.

---

11)(5p)Skriv ett fullständigt program som läser in två namn, byter värden mellan namnen och skriver ut de nya värdena. Bytet ska göras av en funktion med prototypen :

```
void byt(char *asp, char *bsp);
```

Du kan anta att namnen är maximalt 80 tecken långa.

12)(5p)Skriv ett fullständigt program som läser in alla reella mätvärden från textfilen Mess.txt till en dynamiskt allokerad vektor. Antalet mätvärden framgår av det första talet i textfilen. Efter läsning ska programmet normalisera mätvärdena i vektorn genom att dividera alla värden med det största mätvärdet och sedan skriva ut vektorn. Textfilen kan exempelvis se ut som :

```
34
5.45, 5.67, 5.34 . . .
. . . . . .
```

---

13)(5p)Fullborda nedanstående program genom att i main-programmet skriva koden som söker efter en inläst nyckelfastighet enligt kommentarerna. Programmet använder en hashtabell med stackar för kollisionshantering och du ska använda de givna funktionerna :

```
#include ”Fastighet.h”

void into_linkhash(struct link *hashtab[], fastighet f);
/* Stoppar in fastigheten f i hashtab */

struct link *search_linkhash(struct link *hashtab[], fastighet key);
/* Om key finns i tabellen returneras en pekare till länken  annars
   NULL */

void main()
{
   fastighet fv[5] = {{”Lugnet 5”, 1230000, 1560000},
                      {”Ladan 3”,  1500000, 2000000},
                                            ……}};
   fastighet key;
   struct link
   {
       fastighet data;
       struct link *next;
   };

   struct link *hashtabell[10], *kp;

   /* Sätt alla element i hashtabell till NULL */
   /* Stoppa in alla fastigheter från vektorn fv in i hashtabell */
   /* Läs in key */
   /* Sök efter key i hashtabell */
   /* Skriv sökresultat */
}
```

---

14)(5p)I textfilen Fastigheter.txt finns ett antal fastigheter radvis med fastighetsbeteckning, taxeringsvärde och försäljningspris enligt:

```
Ladan5      1230000     1560000
Lugnet6     1456000     1870000
 . . . . . . . . . . . .  . . .
```

Skriv ett program som läser alla fastigheter i filen och stoppar dessa i en kö. Töm sedan kön och skriv ut alla fastigheter som har lägre än 75% förhållande mellan taxeringsvärde och försäljningspris. Du ska även implementera och använda funktionen

```
float tax_i_procent_av_pris(fastighet f);
```

i uppgift 6 ovan.

För hantering av kön ska du använda:

```
/* Specifikation av FIFO-lista – Fifo.h */

#include ”fastighet.h”
typedef fastighet datatyp;

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

void in_fifo(linktyp **fpp, datatyp d);
/* Stoppar in d sist i kön */

datatyp ut_fifo(linktyp **fpp);
/* Tar bort och returnerar data först i kön */
```

---

15)(5p)Binärfilen Fastigheter.dat innehåller ett antal fastigheter enligt uppgift 6 ovan. Skriv ett program som läser alla fastigheter från binärfilen och placerar dessa i en tvåvägslista sorterad efter försäljningspris. Använd jfr_fastigheter i uppgift 6 ovan, som du också ska implementera. Skriv sedan ut de fastigheter som ligger mellan två inlästa priser. För hantering av tvåvägslistan ska du använda:

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

#include ”Fatighet.h”
typedef fastighet 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, 050114

1)
```
sp = calloc(12, sizeof(char));
strcpy(sp, ”Gunnar Joki”);
```

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

3)
```
4
```

4)
```
tp = tp->next;
free(lista);
lista = tp;
```

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

6)
```
#include ”Fastighet.h”
#include <stdio.h>

void las_fastighet(fastighet *fp)
{
   printf("Ge fastighetsbeteckning : ");
   gets(fp->bat);
   printf("Ge fastighets taxeringsvärde : ");
   scanf("%d", &fp->tax);
   printf("Ge fastighetens försäljningspris : ");
   scanf("%d, &fp->pris);
}
```

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

void skriv_fastighet(fastighet f)
{
   printf(”Beteckning : %s\n”, f.bet);
   printf(”Taxeringsvärde : %d\n”, f.tax);
   printf(”Försäljningspris : %d\n”, f.pris);
}
```

8)
```
#include ”Fastighet.h”
#include <conio.h>

void main()
{
   fastighet a, b;

   las_fastighet(&a);
   las_fastighet(&b);
   if (jfr_fastigheter(a, b))
   {
      skriv_fastighet(a);
      skriv_fastighet(b);
   }
   else
   {
      skriv_fastighet(b);
      skriv_fastighet(a);
```

```
   }
   getch();
}
```

9)
```
int udda_paritet(unsigned char uch )
{
   int i, sum = 0;

   for (i = 0; i < 8; i++)
   {
      if (uch & (1  <<  i) != 0)
      {
         sum++;
      }
   }
   return (sum % 2)
}
```

10)
```
#include <conio.h>
#include ”Fasighet.h”

void main()
{
   struct link
   {
      fastighet data;
      struct link *next;
   } *tp;
   float  sum = 0.0;
   int antal = 0;

   tp = lista;
   while(tp != NULL)
   {
      sum += (float)tp->data.tax / tp->data.pris * 100;
      antal++;
      tp = tp->next;
   }
   printf(“Medelvärdet för förhållandet mellan taxeringsvärde och
           försäljningspris =  %.1f \n”, sum/antal);
}
```

11)
```
#include <stdio.h>
#include <conio.h>
#include <string.h>

void byt(char *astr, char *bstr)
{
   char temp[81];

   strcpy(temp, astr);
   strcpy(astr, bstr);
   strcpy(bstr, temp);
}

void main()
{
   char anamn[81], bnamn[81];

   printf("Ge första namnet : ");
   gets(anamn);
   printf("Ge andra namnet : ");
```

```
   gets(bnamn);
   byt(anamn, bnamn);
   printf("Första namnet är efter bytet : %s\n", anamn);
   printf("Andra namnet är efter bytet : %s\n", bnamn);
   getch();
}
```

12)

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

void main()
{
   float *mv, max;
   FILE *tsin;
   int i, antal;

   tsin = fopen("Mess.txt", "rt");
   fscanf(tsin, "%d", &antal);
   mv = calloc(antal, sizeof(float));
   for (i = 0; i < antal; i++)
   {
       fscanf(tsin, "%f", &mv[i]);
   }
   fclose(tsin);

   max = mv[0];
   for (i = 1; i < antal; i++)
   {
      if (mv[i] > max)
         max = mv[i];
   }

   for (i =  0; i < antal; i++)
   {
      mv[i] /= max;
   }

   for (i =  0; i < antal; i++)
   {
      printf("%.2f " , mv[i]);
   }

   free(mv);
   getch();
}
```

13)

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

void main()
{
   fastighet fv[5] = {{”Lugnet 5”, 1230000, 1560000},
                      {”Ladan 3”,  1500000, 2000000},
                                                ……}};
   fastighet key, *kp;
   struct link
   {
       fastighet data;
       struct link *next;
   };
```

```
   struct link *hashtabell[10];

   /* Sätt hashtabellens alla element till NULL */
   for (i=0; i < 10; i++)
      hashtabell[i]= NULL;



   /* Stoppa in alla fastigheter från vektorn fv i hashtabellen */
   for (i=0; i < 5; i++)
      into_linkhash(hashtabell, fv[i]);

   /* Läs in key */
   printf(”Vilken beteckning har sökt fastighet? ”);
   gets(key.bet);

   /* Sök i hashtabellen efter key */
   kp = search_linkhash(hashtabell, key);

   /* Skriv sökresultat */
   if (kp == NULLL)
      printf(”Fastigheten finns ej!”);
   else
      skriv_fastighet(*kp);
   getch();
}
```

14)

```
#include "Fastighet.h"

float tax_i_procent_av_pris(fastighet f)
{
   return (float)f.tax / f.pris * 100;
}

#include <stdio.h>
#include <conio.h>
#include "Fifo.h"


void main()
{
   FILE *tsin;
   linktyp *fp = NULL;
   fastighet f;

   tsin = fopen("Fastigheter.txt","rt");
   while (fscanf(tsin, "%s%d%d", f.bet, &f.tax, &f.pris)!= EOF)
   {
in_fifo(&fp, f);
   }
   fclose(tsin);

   while(fp != NULL)
   {
      f = ut_fifo(&fp);
      if(tax_i_procent_av_pris(f) < 75)
      {
         skriv_fastighet(f);
         printf("Förhållandet = %.1f\n", tax_i_procent_av_pris(f));
      }
```

```
   }
   getch();
}
```

15)

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

int jfr_fastigheter(fastighet f1, fastighet f2)
{
   return f1.pris < f2.pris;
}

void main()
{
   FILE *tsin, *bsin, *bsut;
   headtyp *hp;
   linktyp *lp;
   fastighet f;
   int minpris, maxpris;

   newhead(&hp);
   bsin = fopen("Fastigheter.dat", "rb");
   bsut = fopen("Fastigheter.dat", "wb");
   tsin = fopen("Fastigheter.txt", "rt");
   while (fscanf(tsin, "%s%d%d", f.bet, &f.tax, &f.pris) != EOF)
   {
      fwrite(&f, sizeof(fastighet), 1, bsut);
   }
   fclose(tsin);
   fclose(bsut);

   fread(&f, sizeof(fastighet), 1, bsin);
   while(!feof(bsin))
   {
      newlink(&lp);
      putlink(f, lp);
      insort(lp, hp, jfr_fastigheter);
      fread(&f, sizeof(fastighet), 1, bsin);
   }
   fclose(bsin);

   printf("Ge lägsta försäljningspris :  ");
   scanf("%d", &minpris);
   printf("Ge högsta försäljningspris : ");
   scanf("%d", &maxpris);
   lp = firstlink(hp);
   while (lp != NULL)
   {
      f = getlink(lp);
      if (f.pris >= minpris  && f.pris  <= maxpris)
         skriv_fastighet(f);
```

```
        lp = succlink(lp);
    }
    elimhead(&hp);
    getch();
}
```