ÖREBRO UNIVERSITET

INSTITUTIONEN FÖR TEKNIK

Namn:..... Klass:..... Pnr:.....

Omtentamen i Programmeringsmetodik, 5p, Au2, D1 och E1, 010831

Hjälpmedel	: Inga	
Tid	: 8-13	
Ansvarig lärare	: Gunnar Joki	Tel arb: 303317
		Tel hem: 274825

Svaren till uppgifterna 1-15 ska skrivas på tillgängligt utrymme i detta häfte. Behöver du mera utrymme kan du skriva på baksidan eller på extra papper. Lösningarna till uppgifterna 16-19 ska skrivas på utdelat extra papper med maximalt en uppgift per papper. Skriv namn och personnummer på varje inlämnat extra papper.

Den maximala poängen för respektive uppgift står angiven efter uppgiftens nummer. Totalt kan 40 poäng erhållas. För godkänt krävs ca 20 , för betyget 4 ca 28 och för betyget 5 ca 34 poäng.

Om inget speciellt anges gäller frågorna Borland C .

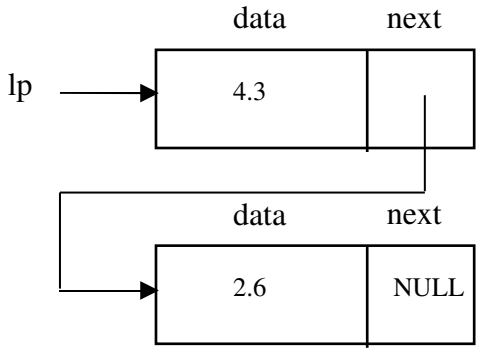
Detta häfte inlämnas.

Lycka till!

1) (1p) Antag att du har en pekare rp som pekar på ett minnesutrymme som innehåller ett reellt tal. Skriv den printf-sats som skriver ut kvadratroten av detta tal med två decimaler.

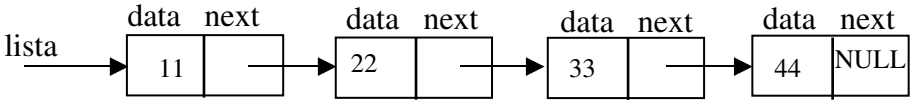
---

2)(1p) Skriv den sats som sätter next för den nedersta länken att peka på den översta länken. Inga extra pekare får definieras.



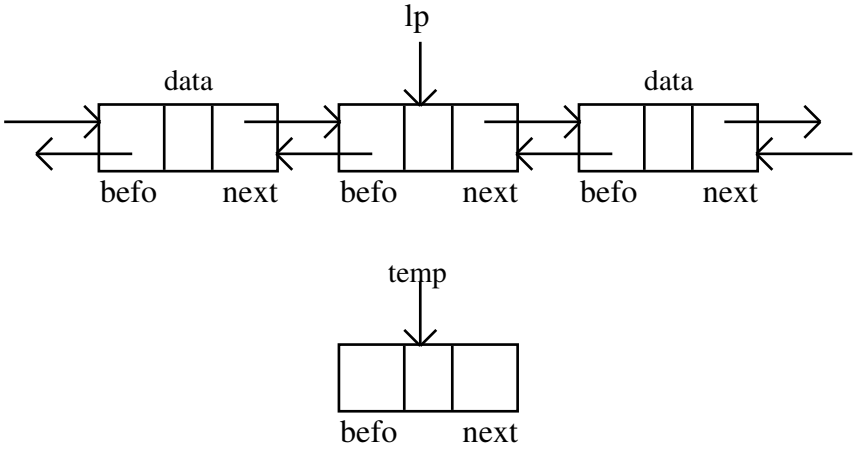
---

3) (1p) Antag att du vill ha in en länk med data lika med 15 efter (till höger) om den länk som har data 11 i nedanstående lista. Skriv de satser som dynamiskt skapar denna länk och stoppar in den i listan vars länkar är av linktyp.



---

4) (1p) Skriv de satser som krävs för att byta plats mellan länken som temp pekar och länken som lp pekar på i nedanstående tvåvägslista. Inga färdiga rutiner eller extra pekare får användas och listan ska hänga ihop efteråt.



5) (1p) Rita det binära träd som skapas då talen 123, 106, 134, 135 och 115 instoppas tal för tal i trädet i den angivna ordningen, om instoppningsfunktionen sätter in data som är mindre i vänsterträdet och som är lika eller större i högerträdet.

---

6)(1p) Som index i en hashtabell för tal kan man exempelvis ta tal % 100. Skissa på hur en sådan hashtabell ser ut då talen ovan i uppgift 5 instoppas i angiven ordning om kollisioner hanteras med öppen adressering och hoppfunktionen hopp = 1 och sedan hopp = hopp + 2.

---

7)(1p) Hur kommer hashtabellen, enligt uppgift 6 ovan att se ut, om man istället använder stackar för att hantera kollisioner?

---

8)(1p) Vad blir utskriften från följande program?

```
#include <stdio.h>
void rf(int n)
{
   if (n > 0)
   {
      rf(n-1);
      printf("%d\n", n);
   }
   printf("%d\n", n);
}

void main()
{
   rf(2);
}
```

---

9)(1p) Antag att du har en 8 bitars unsigned char definierad enligt:

```
unsigned char uch = 5;
```

Ange värdet för uch efter satsen:

```
uch &= 2;
```

---

10)(1p)Antag att du har följande vektor av strängar:

```
char *str[] = {"Hej", "på" ,"dig"};
```

Vad är `*str[1]`?

11)(2p)En tidpunkt under ett dygn kan avbildas som en abstrakt datatyp enligt:

```
/* Specifikation Tid.h */

typedef struct
{
   int tim; /* Timmar 0 - 23 */
   int min; /* Minuter 0 - 59 */
   int sek; /* Sekunder 0 - 59 */
} tid;

void las_tid(tid *tp);
/* Läser tid på formen tim:min:sek */

void skriv_tid(tid t);
/* Skriver tid på formen tim:min:sek */

void andra_tid(int s, tid *tp);
/* Ändrar tid s sekunder */

int innan(tid t1, tid t2);
/* Sant om tidpunkten t1 innan t2, annars falskt */
```

Implementera funktionen las_tid.

---

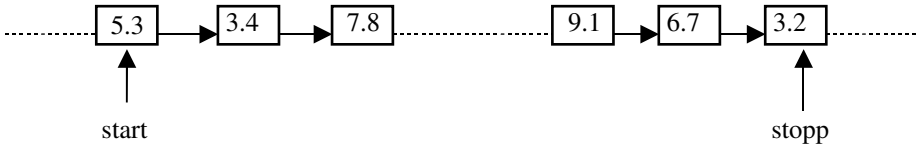
12)(2p)Implementera funktionen skriv_tid, enligt uppgift 11 ovan.

13)(2p)Fullborda funktionen jamnt_tal, som ska returnera sant (1) om uch är ett jämnt tal annars falskt(0). Inga divisionsoperatorer finns tillgängliga för uch, så du måste kontrollera att den minst signifikanta biten är nollställd. Funktionshuvud enligt:

```
int jamnt_tal(unsigned char uch)
{
```

---

14)(2p)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 spridning nedan, som tar start och stopp som parametrar och som returnerar spridningen (största - minsta) för talen från och med start till och med stopp.

```
float spridning(linktyp *start, linktyp *stopp)
{
```

---

15)(2p)Skriv en rekursiv funktion som kopierar strängen str2 till str1. Inga färdiga funktioner får användas. Funktionshuvud enligt:

```
void strkopia(char *str1, char *str2)
{
```

16)(5p)Skriv ett fullständigt program som dynamiskt skapar två reella variabler, läser in värden till dessa och skriver ut värdena i storleksordning med det minsta talet först. Avslutningsvis ska programmet frigöra det dynamiskt allokerade minnet.

---

17)(5p)Skriv ett fullständigt program som upprepat slumpar ensiffriga positiva heltal och lägger dessa på en stack. Upprepningen avslutas då talet 0 slumpas. Programmet ska sedan tömma stacken samtidigt som summan av talen och antalet tal beräknas. Avslutningsvis skrivs medelvärdet av de stackade talen ut. För hantering av stacken ska du använda:

```
/* Specifikation av LIFO-lista -- lifo.h */

typedef int datatyp;     /* Exempelvis */

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 */
```

---

18)(5p)Implementera funktionen andra_tid, enligt uppgift 11 ovan och skriv ett huvudprogram som testar funktionen för en positiv och en negativ ändring.

---

19)(5p)I textfilen tider.txt finns ett antal tider av den abstrakta datatypen tid, enligt uppgift 11 ovan, på formen tim:min:sek radvis enligt:

```
09:12:34
14:08:12
12:45:23
11:55:08
15:23:32
. . . .
. . . .
```

Skriv ett fullständigt program som läser alla tidpunkter från filen och stoppar in dessa i en tvåvägslista sorterad enligt innan-funktionen. Avslutningsvis ska programmet skriva ut de sorterade tiderna radvis på skärmen med rubriker för förmiddagstider resp. eftermiddagstider enligt:

```
Förmiddag
. . . .
09:12:34
11:55:08
. . . .
Eftermiddag
. . . .
12:45:23
14:08:12
15:23:32
```

För hantering av tvåvägslistan ska du använda:

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

#include ”tid.h”
typedef tid 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 omtentamen i Programmeringsmetodik, 5p, 010831

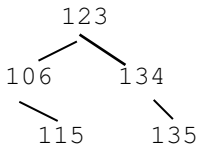
1)
```
printf("%.2f", sqrt(*rp));
```

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

3)
```
linktyp *tp = malloc(sizeof(linktyp));
tp->next = lista->next;
tp->data = 15;
lista->next = tp;
```

4)
```
temp->befo = lp->befo;
temp->next = lp->next;
lp->befo->next = temp;
lp->next->befo = temp;
lp->next = lp->befo = NULL;
```

5)



6)
```
 6 -> 106
15 -> 115
23 -> 123
34 -> 134
35 -> 135
```

7)
```
 6 -> 106
15 -> 115
23 -> 123
34 -> 134
35 -> 135
```

8)
```
0
1
1
2
2
```

9)
```
0
```

10) Tecknet 'p' i strängen "på"

11)

```
#include "Tid.h"
#include <stdio.h>

void las_tid(tid *tp)
{
   printf("Tid på formen 12:23:16 ? ");
   scanf("%2d:%2d:%2d", &tp->tim, &tp->min, &tp->sek);
}
```

12)

```
void skriv_tid(tid t)
{
   printf("Tid = %02d:%02d:%02d\n", t.tim, t.min, t.sek);
}
```

13)

```
int jamnt_tal(unsigned char uch)
{
   if (uch & 1)
      return 0;
   return 1;
}
```

14)

```
float spridning(linktyp *start, linktyp *stopp)
{
   float min, max;
   linktyp *temp;

   temp = start;
   min = max = temp->data;
   while (temp != stopp)
   {
      temp = temp->next;
      if (temp->data > max)
         max = temp->data;
      else if (temp->data < min)
         min = temp->data;
   }
   return max - min;
}
```

15)

```
#include <stdio.h>

void strkopia(char *str1, char *str2)
{
   if (*str2 != '\0')
   {
      strkopia(str1+1, str2+1);
      *str1 = *str2;
   }
   else
      *str1 = '\0';
}
```

16)

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

void main()
{
   float *xp, *yp;

   xp = malloc(sizeof(float));
   yp = malloc(sizeof(float));

   printf("x? ");
   scanf("%f", xp);

   printf("y? ");
   scanf("%f", yp);

   if (*xp < *yp)
      printf("%f\n%f\n", *xp, *yp);
   else
      printf("%f\n%f\n", *yp, *xp);

   free(xp);
   free(yp);
   getch();
}
```

17)

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

void main()
{
   linktyp *lp = NULL;
   int tal, sum = 0, nr = 0;

   randomize();
   tal = random(10);
   while (tal != 0)
   {
      push(&lp, tal);
      tal = random(10);
   }

   while (lp != NULL)
   {
      sum += pop(&lp);
      nr++;
   }

   if (nr > 0)
      printf ("Medel = %.2f", (float)sum / nr);
   else
      printf("Antalet tal är 0!");
```

```
   getch();
}
```

18)

```
void andra_tid(int s, tid *tp)
{
   int totsek = tp->tim * 3600 + tp->min * 60 + tp->sek + s;

   if (totsek < 0)
      totsek += 24 * 3600;
   tp->tim = totsek / 3600 % 24;
   tp->min = totsek % 3600 / 60;
   tp->sek = totsek % 60;
}


/* Huvudprogram -- Tidmain.c */

#include "Tid.h"
#include <conio.h>

void main()
{
   tid t;

   las_tid(&t);
   skriv_tid(t);
   andra_tid(100, &t);
   skriv_tid(t);
   andra_tid(-200, &t);
   skriv_tid(t);
   getch();
}
```

19)

```
/* Tidlista.c */

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

void main()
{
   FILE *tsin;
   headtyp *hp;
   linktyp *lp;
   tid t, m = {12, 0, 0};
   int fm = 1;

   tsin = fopen("tider.txt", "r");
   newhead(&hp);
   while (fscanf(tsin, "%d:%d:%d", &t.tim, &t.min, &t.sek)!= EOF)
   {
      newlink(&lp);
      putlink(t, lp);
      insort(lp, hp, innan);
   }
   fclose(tsin);

   printf("Förmiddag\n");
   lp = firstlink(hp);
   while (lp != NULL)
   {
      t = getlink(lp);
      if (fm && innan(m, t))
      {
         printf("Eftermiddag\n");
         fm = 0;
      }
      skriv_tid(t);
      lp = succlink(lp);
   }
   getch();
}
```