



ÖREBRO UNIVERSITET

INSTITUTIONEN FÖR TEKNIK

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

Omtentamen i Programmeringsmetodik, 5p, D1 och E1, 000901

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.

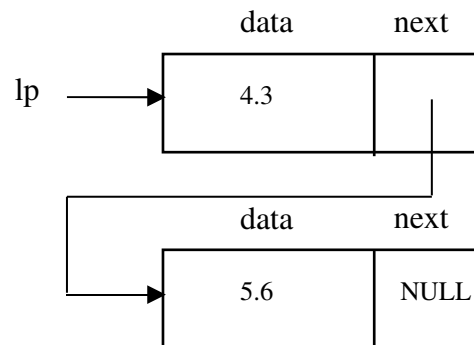
Om inget speciellt anges gäller frågorna Borland C .

Detta häfte inlämnas.

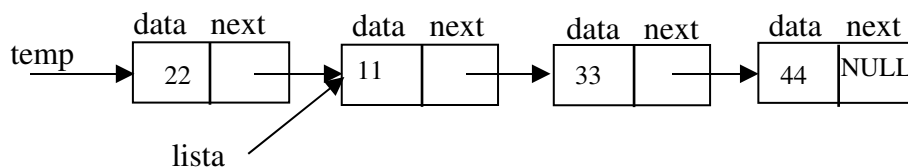
Lycka till!

1) (1p) Antag att du har en pekare `hp` som pekar på ett minnesutrymme som innehåller ett heltal. Skriv den sats som ökar detta heltalsvärde med 1.

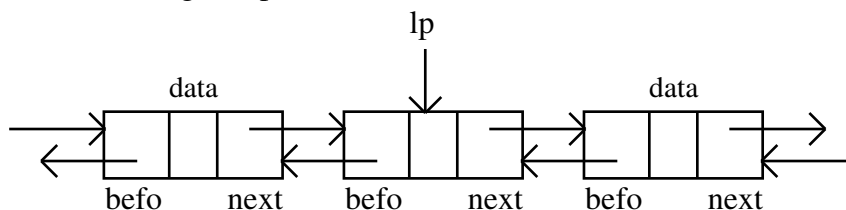
2)(1p) Skriv de satser som tilldelar de två datatermerna de angivna värdena i nedanstående länkade struktur.



3) (1p) Skriv de satser som byter plats mellan länkarna som `lista` resp. `temp` pekar på. Efteråt ska `lista` peka på den första länken (länken längst till vänster) med datavärdet 11 och `temp` på den andra länken som har datavärdet 22.



4) (1p) Skriv de satser som krävs för att ta bort och avallokera 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 bokstäverna i ordet "programmera" instoppas bokstav för bokstav i trädet i den angivna ordningen, om instoppningsfunktionen sätter in data som är mindre i vänsterträdet och lika stora eller större i högerträdet.

6)(1p) Definiera en hashtabell för tecken, som använder sig av öppen adressering för att hantera kollisioner och hashfunktionen tecknets ASCII-kod % 97. Hashtabellens element ska vara pekare till tecken.

7)(1p) Skriv de satser som stoppar in tecknet 'A' med ASCII-koden 65 i en för övrigt tom hashtabell enligt uppgift 6 ovan.

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);
    }
}

void main()
{
    rf(3);
}
```

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

```
unsigned char uch = 3;
```

Ange värdet för uch efter satsen:

```
uch <<= 2;
```

10)(1p) Sätt sp att peka på första tecknet i strängen str och skriv de satser som använder sp för att bestämma strängens längd. Inga färdiga funktioner får användas.

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

11)(2p) Ett simulerat tentamensresultat kan avbildas enligt:

```
/* Specifikation av tentamensresultat - tentres.h */

typedef
struct
{
    char namn[20];          /* Namnet på den som tentat */
    int poang;             /* Antal erhållna poäng */
} tentres;

void las_tentres(tentres *trp);
/* Läser in namn och poäng från tangentbordet. */

void skriv_tentres(tentres tr);
/* Skriver ut namn och poäng på skärmen. */

int jfr_tentres(tentres tr1, tentres tr2);
/* Returnerar sant om tr1 har lägre poäng än tr2. */

char satt_betyg(tentres tr, int gr3, int gr4, int gr5);
/* Returnerar betygen U, 3, 4, 5 utgående ifrån poänggränserna gr3,
gr4 och gr5 */
```

Implementera funktionen skriv_tentres så att utskriften exempelvis blir:

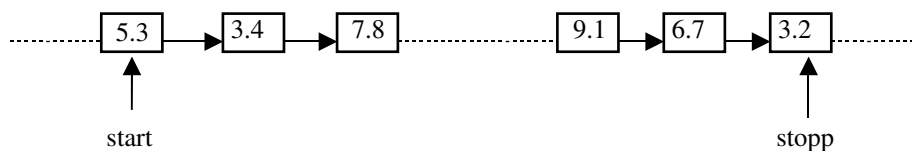
```
Namn : Kurt Olsson
Poäng : 24
```

12)(2p) Implementera funktionen jfr_tentres ovan.

13)(2p) Fullborda funktionen `fler_2_ettor`, som ska returnera sant om antalet ettställda bitar i ASCII-koden för `uch` är fler än 2. Är exempelvis `uch` 'A' med ASCII-koden 01000001 ska funktionen returnera 0 och är `uch` 'C' med ASCII-koden 01000011 ska funktionen returnera 1.

```
int fler_2_ettor(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 `ta_bort_lankar` nedan, som tar `start` och `stopp` som parametrar och som tar bort och avallokerar alla länkar i listan mellan `start` och `stopp`. Start- och stopplänken ska finnas kvar i listan.

```
void ta_bort_lankar(linktyp *start, linktyp *stopp)
{
```

15)(2p) Skriv en iterativ funktion som räknar antalet förekomster av en bokstav i ett binärt träd av tecken enligt uppgift 5 ovan. Funktionshuvud enligt nedan där `nodtyp` är definierad enligt:

```
typedef struct nod
{
    char data;
    struct nod *left, *right;
} nodtyp;
```

```
int antal(nodtyp *np, char bokstav)
{
```

16)(5p) Skriv ett program som skapar en vanlig variabel och en dynamisk variabel av nedanstående posttyp. Programmet ska läsa in data till båda variablerna och sedan byta snabbvalsnummer mellan de två variablerna. Avslutningsvis ska den dynamiska variabeln avallokeras.

```
struct tele
{
    int snabbnr;          /* Snabbval */
    char telnr[15];      /* Telefonnummer */
};
```

17)(5p) Skriv ett program som läser in antalet element och dynamiskt skapar en vektor med detta antal element av den abstrakta datatypen tentres enligt uppgift 11 ovan. Programmet ska fortsätta med att läsa in alla tentamensresultat och sedan skriva ut dessa på skärmen. Avslutningsvis ska vektorn avallokeras.

18)(5p) I textfilen snabbnr.txt finns ett antal snabbvalsnummer med motsvarande telefonnummer enligt:

```
1  -----  019345675
5  -----  019456789
6  -----  019123456
8  -----  081234567
2  -----  082345678
4  -----  019214365
3  -----  019876543
7  -----  016234567
```

Skriv ett program som läser data från filen till en FIFO-lista av poster enligt uppgift 16 ovan och visar alla poster på skärmen. Efter detta ska programmet fråga efter ett snabbvalsnummer och leta efter motsvarande telefonnummer i listan. Finns snabbvalsnumret i listan ska på skärmen stå vilket telefonnummer som rings. Finns inte snabbvalsnumret i listan ska programmet först fråga efter vilket telefonnummer som hör ihop med detta snabbvalsnummer och sedan ringa. Avslutningsvis skrivs den aktuella listan ut på samma fil. För hantering av FIFO-listan ska du använda:

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

typedef struct tele /* Exempelvis */
{
    int snabbnr;      /* Snabbval */
    char telnr[15];  /* Telefonnummer */
} datatyp;

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 */

```

- 19)(5p) I binärfilen tentres.dat finns ett antal tentamensresultat av typen tentres enligt uppgift 11 ovan. Skriv ett program som läser hela filen och stoppar in resultaten sorterade efter poäng i en tvåvägslista. Efter instoppning ska listan genomlöpas och resultatet i form av tentres och betyg skrivs ut. Betyg sätts genom anrop av funktionen satt_betyg i tentres.h, där gränserna är 20, 28 och 34 poäng precis som på tentan som du skriver just nu. Funktionen satt_betyg ska du också implementera. För hantering av tvåvägslistan ska du använda:

```

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

#include "tentres.h"
typedef tentres 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 */

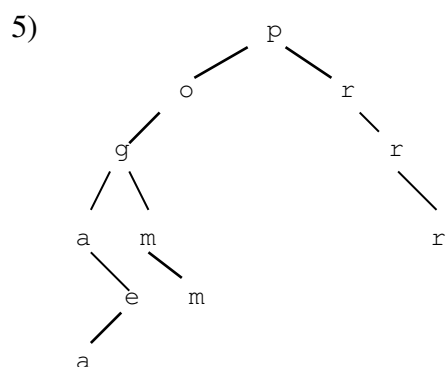
void elimlink(linktyp **lpp);
/* Tar bort, avallokerar och NULL-ställer länken */

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, D1 och E1, 000901

- 1) `*hp = *hp + 1;`
- 2) `lp->data = 4.3;`
`lp->next->data = 5.6;`
- 3) `temp->next = lista->next;`
`lista->next = temp;`
- 4) `lp->befo->next = lp->next;`
`lp->next->befo = lp->befo;`
`free(lp);`
`lp = NULL;`



- 6) `char *hashtabell[97];`
- 7) `hashtabell[65] = malloc(sizeof(char));`
`*hashtabell[65] = 'A';`
- 8) 1
2
3
- 9) 12
- 10) `int len = 0;`

`sp = str;`
`while (*sp != '\0')`
`{`
`sp++;`
`len++;`
`}`

- ```

11) #include "tentres.h"
#include <stdio.h>

void skriv_tentres(tentres tr)
{
 printf("Namn : %s\n", tr.namn);
 printf("Poäng : %d\n", tr.poang);
}

12) #include "tentres.h"

int jfr_tentres(tentres tr1, tentres tr2)
{
 return tr1.poang < tr2.poang;
}

13) int fler_2_ettor(unsigned char uch)
{
 int i, sum = 0;

 for (i = 0; i <= 7; i++)
 {
 if (uch & (1 << i))
 sum++;
 }
 return sum > 2;
}

14) void ta_bort_lankar(linktyp *start, linktyp *stopp)
{
 linktyp *temp;

 temp = start->next;
 while (temp != stopp)
 {
 start->next = temp->next;
 free(temp);
 temp = start->next;
 }
}

15) int antal(nodtyp *np, char bokstav)
{
 int sum = 0;

 while (np != NULL)
 {
 if (bokstav == np->data)
 {
 sum++;
 np = np->right;
 }
 else if (bokstav < np->data)
 np = np->left;
 else
 np = np->right;
 }
 return sum;
}

```

16)

```

/* Huvudprogram -- dyntele.c */
#include <stdio.h>
#include <stdlib.h>

void main()
{
 struct tele
 {
 int snabbnr; /* Snabbval */
 char telnr[15]; /* Telefonnummer */
 } *telepoint, telepost;
 int temp;

 /* Allokera dynamiskt en första post och läs in data */
 telepoint = malloc(sizeof(struct tele));
 printf("Snabbnr : ");
 scanf("%d", &telepoint->snabbnr);
 getchar();
 printf("Telnr : ");
 gets(telepoint->telnr);

 /* Läs in data till en andra post */
 printf("Snabbnr : ");
 scanf("%d", &telepost.snabbnr);
 getchar();
 printf("Telnr : ");
 gets(telepost.telnr);

 /* Byt snabbnr */
 temp = telepoint->snabbnr;
 telepoint->snabbnr = telepost.snabbnr;
 telepost.snabbnr = temp;

 /* Frigör minne */
 free(telepoint);
 telepoint = NULL;
}

```

17)

```

/* Huvudprogram -- tentamen.c */

#include "tentres.h"
#include <stdio.h>
#include <stdlib.h>

void main()
{
 tentres *tv;
 int i, nr;

 /* Skapa en vektor med ett inläst antal tentamensresultat */
 printf("Antal studenter : ");
 scanf("%d", &nr);
 getchar();
 tv = calloc(nr, sizeof(tentres));

 /* Läs tentamensresultat till vektorn */
 for (i = 0; i < nr; i++)
 {

```

```

 las_tentres(&tv[i]);
 }

 /* Visa tentamensresultat på skärmen */
 for (i = 0; i < nr; i++)
 {
 skriv_tentres(tv[i]);
 }

 /* Frigör minne */
 free(tv);
 tv = NULL;
}

```

18)

```

/* Huvudprogram -- snabbtel.c */

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

void main()
{
 FILE *tsinut;
 int slaget_snabbnr;
 datatyp snr;
 linktyp *fp = NULL, *tp;

 tsinut = fopen("snabbnr.txt", "r+");

 /* Läs från fil till lista och skriv snabbnr och telefonnummer */
 while (fscanf(tsinut, "%d ----- %s", &snr.snabbnr, snr.telnr)
 != EOF)
 {
 infifo(&fp, snr);
 printf("%d ----- %s\n", snr.snabbnr, snr.telnr);
 }

 /* Ring */
 printf("Snabbnr : ");
 scanf("%d", &slaget_snabbnr);
 getchar();

 /* Leta efter telefonnummer */
 tp = fp;
 while(tp != NULL && slaget_snabbnr != tp->data.snabbnr)
 {
 tp = tp->next;
 }

 /* Numret finns i listan */
 if (tp != NULL)
 snr = tp->data;
 else
 {
 /* Läs in nytt nr till listan */
 snr.snabbnr = slaget_snabbnr;
 printf("Telnr : ");
 gets(snr.telnr);
 infifo(&fp, snr);
 }
}

```

```

/* Ring */
printf("Ringer : %s\n", snr.telnr);

/* Spara undan listan på fil */
rewind(tsinut);
while (fp != NULL)
{
 snr = utfifo(&fp);
 fprintf(tsinut, "%d ----- %s\n", snr.snabbnr, snr.telnr);
}
fclose(tsinut);
}

```

19)

```

char satt_betyg(tentres tr, int gr3, int gr4, int gr5)
{
 if (tr.poang < gr3)
 return 'U';
 else if (tr.poang < gr4)
 return '3';
 else if (tr.poang < gr5)
 return '4';
 else
 return '5';
}

/* Huvudprogram -- betyg.c */
#include "twolist.h"
#include <stdio.h>

void main()
{
 FILE *bsin;
 headtyp *hp;
 linktyp *lp;
 tentres tr;

 /* Läs data från fil till lista */
 newhead(&hp);
 bsin = fopen("tent.dat", "rb");
 fread(&tr, sizeof(tentres), 1, bsin);
 while (!feof(bsin))
 {
 newlink(&lp);
 putlink(tr, lp);
 insort(lp, hp, jfr_tentres);
 fread(&tr, sizeof(tentres), 1, bsin);
 }
 fclose(bsin);

 /* Skriv ut data och sätt betyg */
 lp = firstlink(hp);
 while (lp != NULL)
 {
 tr = getlink(lp);
 skriv_tentres(tr);
 printf("Betyg : %c\n\n", satt_betyg(tr, 20, 28, 34));
 lp = succlink(lp);
 }
}

```

}