Høgskolen i Gjøvik
Avdeling for teknologi

KONTINUASJONSEKSAMEN

FAGNAVN: **Grunnleggende datakunnskap og programmering**

FAGKODE: **L 182 A**

EKSAMENSDATO: **9. august 2002**

KLASSE(R): **01HIND*, 01HINE*, 01HDMU*, 01HING***

TID: **09.00-13.00**

FAGLÆRER: **Frode Haug**

ANTALL SIDER UTLEVERT: **8 (inkludert denne forside)**

TILLATTE HJELPEMIDLER: **Kun læreboka ”OOP in C++”**

- Kontroller at alle oppgavearkene er tilstede.

- Innføring med penn, eventuelt trykkblyant som gir gjennomslag. Pass på så du ikke skriver på mer enn ett innføringsark om gangen (da det blir uleselige gjennomslag når flere ark ligger oppå hverandre).

- Ved innlevering skilles hvit og gul besvarelse, som legges i hvert sitt omslag.

- Oppgavetekst, kladd og blå kopi beholder kandidaten til klagefristen er over.

- Ikke skriv noe av din besvarelse på oppgavearkene. Men, i oppgavetekst der du skal fylle ut svar i tegning/tabell/kurve, skal selvsagt dette innleveres sammen med hvit besvarelse.

- Husk kandidatnummer på alle ark. Ta vare på dette nummeret til sensuren faller.

Dette eksamenssettet består av tre ulike oppgavetyper:

Oppgave 1 omhandler teori fra datateknikk-delen.
Oppgave 2 omhandler hva som blir utskriften fra noen ulike program.
Oppgave 3 omhandler et litt større programmerings-case.

NB: De tre oppgavene er totalt uavhengige og kan derfor løses separat.

Oppgave 1 (30 %)

a) Internet:

a1) Forklar kort (max. ½ A4-ark) de mest typiske funksjonene/mulighetene i en browser.

a2) Forklar kort (max. ½ A4-ark) de mest typiske funksjonene/mulighetene i et mail-program.

a3) Hva står forkortelsen ”FTP” for ? Forklar kort (max. ½ A4-ark) hva ”FTP” er.

b) Datamaskinen (periferiutstyr / innmat):

b1) Forklar kort (max. ½ A4-ark) om disksystemet (disketter, harddisker).

b2) Forklar kort (max. ½ A4-ark) om DVD.

c) Datasikkerhet:

c1) Nevn fem ulike eksempler på sabotasje ifm. datakriminalitet.

c2) Nevn fem eksempler på “EDB-tekniske sikkerhetstiltak”.

c3) Nevn fem eksempler på *smittefarer* for datavirus (hvordan de manuelt sprer seg).

d) Personvern / etikk:

d1) Hvilke *typer* foretak *måtte* tidligere (etter personregisterloven av 1978) ha konsesjon ?

d2) Nevn fem store hovedkategorier av forskjeller på personopplysningsloven (av 2000) ift. personregisterloven.

d3) Nevn fem etiske problemområder ifm. Internet.

Oppgave 2 (20 %)

a) Hva blir utskriften fra følgende program (litt hjelp: det blir 5 linjer):

```
#include <iostream>

using namespace std;

char txt[] = "SOPPEKOKER ANNE MARI MARIANNE";

int main()  {
  int i, j;

  for (i = 2; i <= 18;  i*=i)
    cout << txt[i] << '\n';

  for (i = 25, j = 2;  i >= 0;  i-=j, j*=i)
    cout << txt[i] << '\n';

  return 0;
}
```

b) Hva blir utskriften fra følgende program (litt hjelp: det blir 5 linjer):

```
#include <iostream>

using namespace std;

struct rec  {
  char text[10];
  char tegn;
};

rec recorder[] =  { {"MOR", 'M'},  {"FAR", 'F'},   {"TANTE", 'T'},
                    {"BROR", 'B'}, {"NIESE", 'N'}, {"BABY", 'B'}   };

void funk0(char txt[], char c)
  {  cout << txt << '\t' << c << '\n';  }

void funk(rec a)
  {  funk0(a.text, a.tegn);  }

void funk(char txt[], char c = 'X')
  {  funk0(txt, c);  }

void funk(char c, char txt[]="BESSA")
  {  funk0(txt, c);  }

void funk(int i, int j)
  {  rec a = recorder[i];  recorder[i] = recorder[j];  recorder[j] = a;  }

int main()  {
  funk(recorder[2]);
  funk(recorder[3].text);
  funk(recorder[1].tegn);
  funk(4, 5);
  funk(recorder[4].text, 'Z');
  funk(recorder[5].tegn, "BESSANT");
  return 0;
}
```

Oppgave 3 (50 %)

NB: Les *hele* teksten for denne oppgaven *nøye*, før du begynner å besvare noe som helst. Studer vedlegget, som inneholder mange viktige opplysninger som du trenger/skal bruke. Legg spesielt merke til, og *bruk*, const'ene, objektenes (ferdiglagde) medlemsfunksjoner, globale variable, de tre overloadede funksjonene "les" og funksjonen "finn".

Innledning

I denne oppgaven skal du lage et lite program som holder orden på en liten boksamling. For hver bok skal det lagres dens tittel og unike ISBN-nummer. En forfatter kan selvsagt ha skrevet flere bøker. Derfor blir opplysningene om en forfatter (navn, adresse og telefon) lagret i egne objekter. Et Bok-objekt vil derfor bare henvise til nummeret for et gitt Forfatter-objekt. Alle dataene om bøkene og forfatterne ligger lagret på to ulike filer. Du skal i denne eksamensoppgaven faktisk *ikke* skrive noe av denne koden.
Du kan bare forutsette at disse dataene er på plass i datastrukturen/hukommelsen, og at de globale variablene "siste_bok" og "siste_forfatter" er satt. Din oppgave er å skrive resten av programmet angitt i vedlegget og denne oppgaveteksten.

Datastrukturen

I vedlegget kan du på den første og øverst på den andre siden se deklarasjoner/definisjoner av const'er, to klasser og globale variable. *Dette skal være alt det du trenger av objekter, medlemsfunksjoner og globale variable for å løse denne eksamensoppgaven !*
Datastrukturen består altså av arrayene "boker" (med Bok-objekter) og av "forfattere" (med Forfatter-objekter). I disse er indeksene f.o.m. nr.1 t.o.m. nr."siste_bok"/"siste_forfatter" i bruk.

Oppgaven

a)

Lag funksjonene void skriv_alle_boker() **og** void Bok :: skriv(int n, bool forf)

Funksjonene går gjennom det som er brukt av arrayen "boker". For hver Bok skrives *en* linje med dets indeks i arrayen (kommer inn som parameter til "Bok::skriv(…)"), tittel og ISBN-nummer.
På en *annen* linje skrives kun navnet til forfatteren. Programmet skal stanse for hver tiende utskrevne bok, og vente på at brukeren skriver ett tegn.
Hint: At forfatternavnet også skal skrives angis vha. variabelen "forf" i "Bok::skriv(…)". Denne
funksjonen bruker "Forfattere::hent(…)" for å få tak i (og dermed kan skrive ut) dette navnet.

b)

Lag funksjonene void skriv_bok() **,** void Bok :: skriv_alt() **og** void Forfatter :: skriv()

Funksjonene ber først brukeren (vha. aktuell "les"-funksjon) om nummeret/indeksen for en Bok. Deretter skrives *alle* data om denne boken ut på skjermen, dvs. dens tittel og ISBN-nummer, samt *alle* forfatterens data (dvs. navn, adresse og telefon).
Hint: Funksjonen "Bok::skriv_alt()" skal nå altså skrive kun tittel og ISBN-nummer, samt at den
kaller aktuelt forfatter-objekt sin "skriv()" for å få skrevet ut *alt* om forfatteren.

c) **Lag funksjonen** void skriv_alt_av_forfatter()

Funksjonen ber først brukeren (vha. aktuell ”les”-funksjon) om en forfatters navn. Om denne forfatteren *ikke* finnes (bruk funksjonen ”finn”) så kommer det en melding om det. I motsatt fall gås det gjennom hele ”boker”-arrayen, og indeksnummeret, tittel og ISBN-nummer for alle bøkene av denne forfatteren skrives ut på hver sin linje på skjermen.
Hint: Bruk funksjonene ”Bok::skrevet_av(…)” og Bok::skriv(…, false)”.
NB: Du trenger her *ikke* å stanse for hver tiende/tyvende bok.

d) **Lag funksjonen** void slett_bok()

Funksjonen ber først brukeren (vha. aktuell ”les”-funksjon) om nummeret/indeksen for en Bok. Boken fjernes så rett og slett ved å flytte den siste boken i ”boker” inn på dennes plass i arrayen. Husk å telle ned den globale variabelen ”siste_bok”!

e) **Lag funksjonen** void ny_bok() **,** void Bok :: les_data(…) **og** void Forfatter :: les_data(…)

Funksjonene sjekker først om det er plass til flere bøker og forfattere i datastrukturen. Om så *ikke* er tilfelle, skrives det en melding om det. I motsatt fall så ber man brukeren (vha. aktuell ”les”-funksjon) om forfatterens navn. Det undersøkes så om denne forfatteren allerede finnes (vha. funksjonen ”finn”). Om så *ikke* er tilfelle tilkalles den nye forfatterens ”les_data()”, med navnet som parameter. Denne funksjonen leser også forfatterens adresse og telefon (vha. aktuell ”les”).
I begge tilfellene (forfatteren finnes allerede eller er ny) blir bokens tittel og ISBN lest inn vha. den nye bokens ”les_data(…)” med forfatterens aktuelle nummer som parameter.
NB: Husk evt. å telle opp variablene ”siste_bok” og ”siste_forfatter”.

f) **Lag funksjonen** void slett_forfatter()

Funksjonen ber først om en forfatters navn (vha. aktuell ”les”). Om vedkommende *ikke* finnes (bruk ”finn”), så kommer det en melding om det. I motsatt fall så undersøkes det (vha. for-løkke og ”Bok:skrevet_av(…)”) om noen av Bok-objektene henviser til denne forfatteren. Gjør noen det, så kommer det også en melding, og forfatteren slettes/fjernes *ikke*. Om *ingen* henviser, så flyttes siste forfatter inn på plassen til den som skal slettes. Alle Bok-objekter som evt. henviste til den aller siste forfatteren, blir så oppdatert (vha. ”Bok::skrevet_av(...)” og ”Bok::sett(…)”) med det nye nummeret til forfatteren. Husk å telle ned variabelen ”siste_forfatter”!

Klargjøring og forutsetninger

- Programmet fungerer altså slik (ut fra **oppgave 3e**) at ingen duplikate forfattere vil forekomme, men det kan gjerne legges inn duplikate bøker (tittel og/eller ISBN-numre).
- Vi forutsetter at *enhver* bok bare har *en* forfatter. Det er heller ikke mulig å endre/editere på en forfatters adresse eller telefon-nummer (noen småkritiske mangler kanskje …..).
- En bok slettes (**oppgave 3d**) helt uavhengig av ”forfattere”-arrayen. Dette medfører at det kan bli liggende ikke-refererte forfattere i den. Dette er helt OK, da indeksen evt. blir tatt i bruk igjen om en ny bok (**oppgave 3e**) har vedkommende som forfatter. **NB:** Det finnes ingen funksjon i det nåværende programmet som går gjennom og viser alle (evt. ikke-refererte) forfattere.
- Du trenger *ikke* å lage noen sjekk på gyldigheten av ISBN- og telefon-numre.
- Gjør dine egne forutsetninger dersom du finner oppgaveteksten upresis eller ufullstendig. Gjør i så fall rede for disse forutsetningene *først* i besvarelsen din.

Lykke til !

frode@haug.com

Vedlegg: Halvferdig programkode (.tpl-fil)

```
                        //  INCLUDE:
#include <iostream>             //  cin, cout
#include <iomanip>              //  setw
#include <cstring>              //  strlen, strcmp, strcpy
#include <cctype>               //  toupper
using namespace std;

                        //  CONST:
const int MAX_BOK   = 200;      //  Max. antall titler i datastrukturen.
const int MAX_FORF  =  50;      //  Max. antall forfattere i datastrukturen.
const int STRLEN    =  40;      //  Max.lengde for en tekststreng.

                        //  KLASSER:
class Bok  {
  private:                      //  Data om en bok:
    char tittel[STRLEN];        //   - Tittel.
    char ISBN[STRLEN/2];        //   - ISBN-nummer.
    int  forf_nr;               //   - Nummer i arrayen "forfattere".

  public:
    void skriv(int n, bool forf);       //  Lag innmat ifm. oppgave 3a  (og 3c)
    void skriv_alt();                   //  Lag innmat ifm. oppgave 3b
    void les_data(int nr);              //  Lag innmat ifm. oppgave 3e
    void sett(int nr)           {  forf_nr = nr;  }
    bool skrevet_av(int nr)     {  return (forf_nr == nr);  }
};


class Forfatter  {
  private:                      //  Data om en forfatter:
    char navn[STRLEN];          //   - Navn
    char adr[STRLEN];           //   - Adresse (gate, poststed, land).
    char tlf[STRLEN/2];         //   - Telefon (inkl. evt. landskode).

  public:
    void skriv();                       //  Lag innmat ifm. Bok::skriv_alt()
    void les_data(char nvn[]);          //  Lag innmat ifm. oppgave 3e
    void hent(char nvn[])       {  strcpy(nvn, navn);       }   // Returnerer navn.
    bool er_lik(char t[])       {  return (!strcmp(navn, t)); }// Sammenligner.
};

                        //  DEKLARASJON AV FUNKSJONER:
void  skriv_meny();                         void skriv_alle_boker();        //  Oppgave 3a
char  les();                                void skriv_bok();               //  Oppgave 3b
void  les(char t[], char s[], int LEN);     void skriv_alt_av_forfatter();  //  Oppgave 3c
int   les(char t[], int min, int max);      void slett_bok();               //  Oppgave 3d
int   finn(char t[]);                       void slett_forfatter();         //  Oppgave 3f
                                            void ny_bok();                  //  Oppgave 3e
```

```
                        //  GLOBALE VARIABLE:
Bok       boker[MAX_BOK+1];              //  Array av Bok-objekter.
Forfatter forfattere[MAX_FORF+1];                //  Array av Forfatter-objekter.
int  siste_bok,                          //  Aktuelt antall bøker i datastrukturen.
     siste_forfatter;                    //  Aktuelt antall forfattere i datastrukturen.


int main()  {              //  HOVEDPROGRAMMET:
  char valg;

skriv_meny();
  valg = les();
  while (valg != 'Q')  {
    switch(valg)  {
      case 'N':  ny_bok();                 break;          //  Oppgave 3e
      case 'A':  skriv_alle_boker();       break;          //  Oppgave 3a
      case 'B':  skriv_bok();              break;          //  Oppgave 3b
      case 'F':  skriv_alt_av_forfatter();         break;          //  Oppgave 3c
      case 'T':  slett_bok();              break;          //  Oppgave 3d
      case 'S':  slett_forfatter();        break;          //  Oppgave 3f
      default :  skriv_meny();             break;
    }
    valg = les();
  }
  cout << "\n\n";
  return 0;
}
                        //  DEFINISJON AV FUNKSJONER:
void skriv_meny()  {           //  Skriver brukerens lovlige valg:
  cout << "\nFølgende kommandoer er lovlige:";
  cout << "\n\tN - Ny bok";
  cout << "\n\tA - skriv data om Alle bøkene";
  cout << "\n\tB - skriv alle data om en Bok";
  cout << "\n\tF - skriv alle data om en Forfatter og alle dens bøker";
  cout << "\n\tT - sletT en bok";
  cout << "\n\tS - Slett (om mulig) en forfatter";
  cout << "\n\tQ - Quit / avslutt";
}

char les()  {                  //  Leser, upcaser og returnerer ETT tegn:
  char ch;
  cout << "\n\nKommando:  ";
  cin >> ch;  cin.ignore();      //  Leser og forkaster neste tegn (='\n').
  return (toupper(ch));          //  Upcaser og returnerer.
}

void les(char t[], char s[], int LEN) {          //  Leser inn en ikke-blank tekst:
  do  {
    cout << '\t' << t << ": ";   //  Skriver ledetekst.
    cin.getline(s, LEN);         //  Leser inn tekst.
  } while (strlen(s) == 0);      //  Sjekker at tekstlengden er ulik 0.
}
```

```
int les(char t[], int min, int max)  {    // Leser et tall i et visst intervall.
  int n;
  do  {                                   // Skriver ledetekst:
    cout << '\t' << t << " (" << min << '-' << max << "): ";
    cin >> n;                             // Leser inn ett tall.
  } while(n < min  ||  n > max);          // Sjekker at i lovlig intervall.
  return n;                               // Returnerer innlest tall.
}

                                  //  Ut fra en forfatters navn, så returneres vedkommendes indeks i
int finn(char t[])  {             //        i "forfattere",evt. '0' om han/hun ikke finnes.
  for (int i = 1;  i <= siste_forfatter;  i++)
     if (forfattere[i].er_lik(t))  return i;
  return 0;
}

void skriv_alle_boker()  {
        //  Oppgave 3a:  Lag innmaten
}

void skriv_bok()  {
        //  Oppgave 3b:  Lag innmaten
}

void skriv_alt_av_forfatter()  {
        //  Oppgave 3c:  Lag innmaten
}

void slett_bok()  {
        //  Oppgave 3d:  Lag innmaten
}

void ny_bok()  {
        //  Oppgave 3e:  Lag innmaten
}

void slett_forfatter()  {
        //  Oppgave 3f:  Lag innmaten
}
```