Høgskolen i Gjøvik

Avdeling for Teknologi

E K S A M E N

FAGNAVN:	**Grunnleggende datakunnskap og programmering**
FAGNUMMER:	**L 182 A**
EKSAMENSDATO:	**7. desember 2000**
KLASSE:	**00HINDA / 00HINDB / 00HINEA 00HDMUA / 00HDMUB / 00HINGA**
TID:	**09.00-13.00**
FAGLÆRER:	**Frode Haug**

ANTALL SIDER UTLEVERT: **7 (inkludert denne forside)**

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

- Kontroller at alle oppgavearkene er tilstede.
- Innføring med penn, evt. trykkblyant som gir gjennomslag. Pass på at du ikke skriver på mer enn ett innføringsark om gangen (da det blir uleselige gjennomslag om flere ark ligger oppå hverandre når du skriver).
- Ved innlevering skilles hvit og gul besvarelse og legges i hvert sitt omslag. Oppgavetekst, kladd og blå kopi beholder kandidaten.
- Ikke skriv noe av din besvarelse på oppgavearkene.
- Husk kandidatnummer på alle ark.

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 et mail-program.

a2) Forklar kort (max. ½ A4-ark) hva "News/Usenet" er.

a3) Hva står forkortelsen URL for ? Og forklar kort hva det er.

a4) Hva er "Bookmarks"/"Favorites" ?

b) Datamaskinen (periferiutstyr / innmat):

b1) Nevn fem viktige elementer/komponenter som er på datamaskinens hovedkort. Forklar kort (2-3 setninger) om hver av dem.

b2) Redegjør kort om (max. ½ A4-ark pr. punkt):

b2-a) Monitor/skjerm/display
b2-b) DVD
b2-c) Bussen

c) Datasikkerhet:

c1) Nevn fem eksempler på "EDB-tekniske sikkerhetstiltak".

c2) Nevn fem eksempler på "smittefarer" for datavirus (hvordan de manuelt sprer seg).

c3) Hvilke seks hovedområder har vi for "håndtering av virus-problematikken" ?

d) Personvern / etikk:

d1) Hvilke typer foretak har så langt *måttet* ha konsesjon ?

d2) Nevn fem store hovedkategorier av forskjeller på personopplysningsloven (av 2000) ift. personregisterloven (av 1978).

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;

int main()  {
  char txt[] = "MISSISSIPPI";
  int i = 5, j = 6;

  while (i > 0) {
    cout << txt[i] << ' ' << txt[j] << ' '  // Eller:
         << char('K' - 2*i)     << ' '      // static_cast <char>('K'-2*i)
         << char('D' + 2*j - 7) << '\n';    // static_cast <char>('D'+2*j-7)
    i--;  j++;
  }
  return 0;
}
```

b) Hva blir utskriften fra følgende program (litt hjelp: det blir 3 linjer):

```
#include <iostream>
#include <cstring>
using namespace std;

struct rec1  {
  int x, y;
};

struct rec2  {
 rec1 r1, r2;
 char txt[5];
};

rec2 samling[] = { { {100, 200}, {300, 400}, "321" },
                   { {100, 300}, {200, 400}, "231" },
                   { {200, 300}, {100, 400}, "132" } };

rec2 beregn(rec2 rr1, rec2 rr2)  {
  int x1 = (rr1.r1.x + rr2.r1.x) / 2,
      y1 = (rr1.r1.y + rr2.r1.y) / 2,
      x2 = (rr1.r2.x + rr2.r2.x) / 2,
      y2 = (rr1.r2.y + rr2.r2.y) / 2;
  rec2 rr3 = { {x1, y1}, {x2, y2}, ""};  strcpy(rr3.txt, rr2.txt);
  return rr3;
}


int main()  {
  rec2 r3;
  r3 = beregn(samling[0], samling[1]);
  cout << '\n' << r3.r1.x << ' ' << r3.r1.y
       << ' '  << r3.r2.x << ' ' << r3.r2.y << ' ' << r3.txt;
  r3 = beregn(samling[0], samling[2]);
  cout << '\n' << r3.r2.x << ' ' << r3.r2.y
       << ' '  << r3.r1.x << ' ' << r3.r1.y << ' ' << r3.txt;
  r3 = beregn(samling[2], samling[1]);
  cout << '\n' << r3.r1.x << ' ' << r3.r1.y
       << ' '  << r3.r2.x << ' ' << r3.r2.y << ' ' << r3.txt;
  return 0;
}
```

Oppgave 3 (50 %)

NB: Les *hele* teksten for denne oppgaven *nøye*, før du begynner å besvare noe som helst. Studer vedlegget nøye, som inneholder mange viktige opplysninger som du trenger/skal bruke. Legg spesielt merke til (og bruk) alle const'ene, klassene og de globale variablene.

Innledning

I denne oppgaven skal du lage et lite program som holder orden på ulike spilleres poenger i spillet Yatzy.
Spillet som spilles er "tvungen tradisjonell Yatzy". Dette spillet har følgende poster: 1'ere, 2'ere, 3'ere, 4'ere, 5'ere, 6'ere, Sum (for alle 1'erne til 6'erne), Bonus, 1 par, 2 par, 3 like, 4 like, Liten Straight (alle terningverdiene fra 1 til 5), Stor Straight (alle terningverdiene fra 2 til 6), Hus (2+3 like), Sjanse (høyest mulig poensum, uansett terningverdier), Yatzy (5 like) og Totalsum.
"Tvungen Yatzy" spilles på følgende måte: Hver av spillerne som er med kaster etter tur opptil seks terninger max. tre ganger. For hver gang terningene kastes legger man til side (og kaster ikke) de terningene man vil spare på. Først er poenget å få flest mulig 1'ere. Når en spiller er ferdig med sine opptil tre kast, noteres den poengsummen vedkommende har oppnådd på 1'erne. Så er det neste spillers tur til å få flest mulig 1'ere. Når alle spillerne er ferdig med dette, skal de etter tur få flest mulig 2'ere, deretter 3'ere, osv..... Slik fortsetter spillet, post for post som angitt ovenfor. Den som til slutt har oppnådd høyest poengsum, har vunnet spillet.
Din oppgave her blir å skrive resten av programmet angitt i vedlegget og denne oppgaveteksten. Programmet skal altså håndtere hver enkelt spillers poeng på de ulike postene, beregne summer, bonuser og skrive resultatet til fil.

Datastrukturen

I vedlegget kan du på den første siden se deklarasjoner/definisjoner av const'er, to klasser og globale variable (*ett* Poster-objekt, en array med "Spiller"-objekter og to int'er). *Dette skal være alt det du trenger av slike globale variable for å løse denne eksamensoppgaven !*
Inni objektet "postnavnene" ligger den to-dimensjonale char-arrayen "postnavn". Denne inneholder navnet/tittelen for alle postene. Disse ligger f.o.m. indeks nr.0 t.o.m. nr. "siste_post".
Datastrukturen består av arrayen "deltager" som er bygd opp av "Spiller"-objekter. Indeksene fra 1 til "ant_spillere" er i bruk av denne arrayen. Inni hvert enkelt "Spiller"-objekt er det en array: "poeng". Innholdet i nr.'i' refererer til hvilke poeng vedkommende spiller har fått på post nr.'i'.

Under løsningen av denne eksamensoppgaven *skal* du bruke denne datastrukturen !

Oppgaven

a) **Tegn datastrukturen.** Gjør dette detaljert, pent og klart.

b) **Lag funksjonen** int summer(int p[MAX_POSTER], int n, int m)

Funksjonen tar imot en array 'p' og beregner summen av alle elementene f.o.m. indeks nr.'n' t.o.m. indeks nr.'m'. Den returnerer dette svaret.

c) **Lag funksjonene** void initier_spillere() og Spiller:initier(…)

Først skal det spørres om og leses inn i den globale variabelen "ant_spillere". Det *skal* sørges for at denne verdien blir i intervallet 1 - MAX_SPILLERE. Deretter skal det , for hvert aktivt Spiller-objekt, spørres om og leses inn vedkommendes navn. (Du trenger *ikke* å sjekke om navnet kun består av Enter/blanke tegn.) Hele poeng-arrayen til hver spiller skal også bli satt til 0 (null).
NB: Husk at vi bruker indeks nr.1 og oppover for spillerne i arrayen "deltager".

d) **Lag funksjonen** void Poster::les_fra_fil()

Hver enkelt linje på filen "YATZY.DTA" inneholder ett eller flere ord som er teksten for de ulike postene angitt i innledningen. Funksjonen skal lese en og en linje fra denne filen, og legge de leste tekstene inn i arrayen "postnavn" (startende med indeks nr.0). Funksjonen må også passe på at den globale variabelen "siste_post" blir satt til korrekt verdi.

e) **Lag funksjonene** void skriv_resultat_til_fil() og Spiller::skriv_til_fil(…)

Alle spillernes resultater skal skrives til filen "YATZY.RES". For hver enkelt spiller skrives vedkommendes navn (på *en* linje), og deretter, på *hver sin linje*: postnavn og det tallet/poengene vedkommende har på den aktuelle posten.
Hint: Spiller::skriv_til_fil(…) kaller/bruker funksjonen Poster:hent(…)
NB: *Alle* postene og deres tilhørende tall skal skrives, også "Sum", "Bonus" og "Totalsum".

f) **Lag resten av innmaten i** main , Spiller::les_poeng(…) og Spiller::beregn_poengsum(…)

Det gås gjennom alle postene i Yatzy ("siste_post"-1 stk.). For hver post gås det gjennom alle de "ant_spillere" spillerne som er med. For hver poengsum som leses inn (*skal* være et tall mellom 0 og "MAX_POENG") i en spillers "poeng"-array, så skrives aktuell spillers navn og aktuelt postnavn (bruk også her funksjonen Poster:hent(…)) ut på skjermen.
Når poengsummen for 6'erne (posten med indeks nr.5) er innlest for en spiller, skal verdien i post nr.6 ("Sum") automatisk beregnes og skrives ut sammen med postnavnet ("Sum"). Dette er summen av det som ligger i indeks nr.0 til 5 for vedkommende spiller. Om denne summen er større eller lik "BONUS_GRENSE" skal det inn i post nr.7 (også automatisk) legges verdien "BONUS", ellers vil den fortsatt bare være 0 (null). Husk å hoppe over disse to postene (nr.6 og 7) når du videre leser inn spillernes poengsummer !
Når *alle* spillernes poengsummer er innlest skal deres "Totalsum" oppdateres. Dvs. inn i hver enkelt spillers post nr. "siste_post" skal summen av tallene i indeks nr.6 til "siste_post"-1 bli lagt. (Post nr.0-5 skal *ikke* være med i denne summen, da deres sum allerede ligger i post nr.6!)
Hint: Ved summering av del-arrayer (post nr.6 og "siste_post"), bruk funksjonen fra oppgave 3b.

Klargjøring og forutsetninger

- Du trenger ikke å ta hensyn til at:
  - det skal være mulig å korrigere/endre en tidligere angitt poengsum.
  - poengene som skrives inn er fornuftige, f.eks: at '7' ikke er mulig på 6'ere, at 17 er umulig på Liten Straight, eller at 21 er umulig på 4 like.
- 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 <fstream>            //  ifstream, ofstream
#include <cstring>            //  strcpy
#include <iomanip>            //  setw

using namespace std;

                        //  CONST:
const int TXT_LEN           = 20;    //  Max. lengde på tekster (navn).
const int MAX_SPILLERE  =  6;    //  Max. antall spillere som kan delta.
const int MAX_POSTER    = 23;    //  Max. antall ulike poster i Yatzy.
const int MAX_POENG     = 50;    //  Max. antall poeng i EN omgang.
const int BONUS_GRENSE = 63;    //  Om summen av post nr.0-5 (1'ere-6'ere)
const int BONUS             = 50;    //   >= BONUS_GRENSE gis 'BONUS' ekstrapoeng.

                        //  KLASSER:
class Poster {                        //  Klasse med postnavnene:
  private:
    char postnavn[MAX_POSTER][TXT_LEN];     //  Navnet/tittelen til de ulike postene.

  Public                              //  Returnerer navnet til post nr.'n':
    void hent(char navn[], int n)  {  strcpy(navn, postnavn[n]);  }
    void les_fra_fil();               //  Lag innmat ifm. oppgave 3d
};

class Spiller  {
  private:
    char navn[TXT_LEN];               //  Spillerens navn.
    int  poeng[MAX_POSTER];           //  Hver enkelt post's poeng (nr.0 brukes !)

  public:
    void initier(int n);              //  Lag innmat ifm. oppgave 3c
    void les_poeng(int n);            //  Lag innmat ifm. oppgave 3f
    void beregn_totalsum();           //  Lag innmat ifm. oppgave 3f
    void skriv_til_fil(ostream* ut);  //  Lag innmat ifm. oppgave 3e
};


                        //  DEKLARASJON AV FUNKSJONER:
int  summer(int p[MAX_POSTER], int n, int m);   //  Oppgave 3b
void initier_spillere();                        //  Oppgave 3c
void skriv_resultat_til_fil();                  //  Oppgave 3e


                        //  GLOBALE VARIABLE:
Poster postnavnene;                     //  Alle postnavnene lagres i ETT objekt.
Spiller deltager[MAX_SPILLERE+1];       //  Deltagende spillere (nr.0 brukes ikke).
int  siste_post, ant_spillere;          //  Aktuelt antall poster og spillere.
```

```
int main()  {              //  HOVEDPROGRAMMET:
  int i, j;

  postnavnene.les_fra_fil();          //  Leser postnavene/-titlene fra fil.

  initier_spillere();                 //  Initierer/nullstiller spillerne.

        //  Oppgave 3f:  Lag resten av innmaten

  skriv_resultat_til_fil();           //  Skriver alt til fil.

  return 0;
}

                        //  DEFINISJON AV KLASSE-FUNKSJONER:
void Poster::les_fra_fil()  {
        //  Oppgave 3d:  Lag innmaten
}

void Spiller::initier(int n)  {
        //  Oppgave 3c:  Lag innmaten
}

void Spiller::les_poeng(int n)  {
        //  Oppgave 3f:  Lag innmaten
}

void Spiller::beregn_totalsum()  {
        //  Oppgave 3f:  Lag innmaten
}

void Spiller::skriv_til_fil(ostream* ut)  {
        //  Oppgave 3e:  Lag innmaten
}

                        //  DEFINISJON AV FUNKSJONER:
int  summer(int p[MAX_POSTER], int n, int m)  {
        //  Oppgave 3b:  Lag innmaten
}

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

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