NTNU
Kunnskap for en bedre verden

Institutt for datateknologi og informatikk

Kontinuasjonseksamensoppgave i PROG1003 – Objekt-orientert programmering

Faglig kontakt under eksamen:	**Frode Haug**
Tlf:	**950 55 636**

Eksamensdato:	**12.august 2021 - HJEMME (A-F)**
Eksamenstid (fra-til):	**09:00-13:00 (4 timer)**
Hjelpemiddelkode/Tillatte hjelpemidler:	**F - Alle trykte og skrevne.** (kalkulator er *ikke* tillatt)

Annen informasjon:

Målform/språk:	**Bokmål**
Antall sider (inkl. forside):	**6**

Informasjon om trykking av eksamensoppgaven

Originalen er:

1-sidig X **2-sidig** ☐

sort/hvit X **farger** ☐

Skal ha flervalgskjema ☐

Kontrollert av:

_____

Dato Sign

NB: Oppgave 1a, 1b, 1c, 1d, 2 og 3 er totalt uavhengige og kan derfor løses separat.

Oppgave 1 (40%)

Vi har følgende konvertering/sammenheng mellom siffer/tall og bokstav:

0 = A 1 = B 2 = C 3 = D 4 = E 5 = F 6 = G 7 = H 8 = I 9 = J

a)

Vi har:

- `string tekst = "....."`;

  Der `"....."` totalt er på 30 tegn. Den er bygd opp av *ett* ord på seks bokstaver, gjentatt fem ganger rett etter hverandre. Seks-bokstavs ordet (norsk eller engelsk) skal du selv velge/finne. Det *skal* starte med bokstaven som tilsvarer det *siste* sifferet i kandidatnummeret ditt. Ordet *må* inneholde *minst* fire ulike bokstaver. **F.eks.** er det siste sifferet '5', altså bokstaven 'F' (se ovenfor), velges ordet «FRISKT». `tekst` blir da: `"FRISKTFRISKTFRISKTFRISKTFRISKT"` (*ikke* lov å velge dette ordet - for de med det siste sifferet lik '5'!)
- `int s1 = <siste siffer i kandidatnummeret>;`
- `int s2 = <nest siste siffer i kandidatnummeret>;`
- `int i = tekst.length() - 5 - s1;`
- `int j = ( s2 % 4 ) + 2;`

Skriv opp 6-bokstavsordet.
Hva vil en løkke som starter på `i`, for hver gang *minsker* med `j`, går *fire* ganger og inneholder koden: `cout << ' ' << tekst[i];` skrive ut?
Hva skrives ved: `cout << tekst.substr(5+s1+s2, 4); ?`

b)

Vi har:

- en tom `<list>` med `int`
- `int siffer = <kandidatnummeret> % 10;`

En for-løkke med løkkevariabel `i` går f.o.m. 10 og opp t.o.m. 20 + `siffer`.
Er `i` et partall (10, 12, 14, 16, ….) legges `i` forrest i listen, ellers legges den bakerst.

Hva er tallet/verdien som nå er nr:

- **2, 5 og 8 *forfra* i listen**
- **2, 5 og 9 *bakfra* i listen**

c) (Dette er *ikke* et komplett kompilerende program):

```
1    int main()  {
2       int i, nr;

3       string text = "mor";
4       string tekst(10, '-');

5       cout << tekst.length() << ' ' << tekst.size()
6            << ' ' << tekst.capacity() << '\n';

7       tekst += text + "far" + '!';

8       cout << tekst[2] << tekst[4]
9            << tekst.at(12) << tekst.at(15) << '\n';

10      text = tekst.substr(10, 6);

11      tekst.erase(0, 10);

12      tekst.insert(3, text);

13      tekst.append(3, '.');

14      nr = tekst.find(text);

15      if (text <= tekst.substr(3, 6)) cout << "Ja\n";

16      for (i = 1;  i < 22;  i+=2) tekst.insert(i, " ");

17      while ((nr = tekst.find(' ')) >= 0)
18           tekst.replace(nr, 1, "_");

19      nr = tekst.find_first_not_of("mor_");

20      return 0;
     }
```

Kommenter med *egne ord* hva *hver eneste linje* gjør (henvis til linjenumrene 1-20).

d) **Forklar med *egne ord* begrepene** (max. fire linjer pr. pkt):

1. Standard/default parametre/argumenter
2. Arv
3. Abstrakt baseklasse
4. Selvlaget copy-constructor
5. Direkteposisjonering på fil

Oppgave 2 (40%)

Før du begynner å skrive kode: les *hele* teksten *nøye* for denne oppgaven.
Husk på, og bruk, funksjonene på `LesData2.h`**.**

Du skal lage deler av et program som holder orden på de som besøker et sted (butikk/restaurant/pub/ bibliotek/idrettsarena/teatersal/kino/treningsstudio/svømmehall/….) ifm. smitteverntiltak.

Vi forutsetter at følgende allerede er på plass/kodet:

- alle nødvendige includer
- en «tradisjonell» `main` som tilkaller funksjonene i den ene klassen (pkt.2) nedenfor

Lag *komplett kode* for (eksakt navn på klasser og funksjoner bestemmer du selv):

1. **En klassen for *den som besøker* et sted**. Den *skal* inneholde:
   - fire `private` datamedlemmer: mobiltelefonnummer (som er dens ID/key), antall personer vedkommende har med seg (inkludert en selv), vedkommendes navn og mailadresse
   - en constructor som tar mobilnr som parameter og oppdaterer datamedlemmet med dette
   - en funksjon som returnerer dens ID/key
   - en funksjon som leser inn fra tastaturet verdier til de tre andre datamedlemmene
   - en funksjon som tar et filobjekt som parameter, og skriver *alle* datamedlemmene til denne filen (på *en* linje, på ett eller annet selvvalgt format, men at mobilnr kommer aller først)

   Vi forutsetter at det også er laget en `skrivData()` som skriver *alle* klassens datamedlemmer på ett eller annet format ut på skjermen.

2. **En selvlaget *container*-klasse med alle de som for tiden besøker *ett* sted**. Den *skal* inneholde:
   - to `private` datamedlemmer: stedets navn og en `map` med de som for øyeblikket besøker/ er på stedet. Keyen i `map`en er vedkommendes mobilnr, og det pekes til den besøkende.
   - en constructor som tar navnet som parameter og oppdaterer datamedlemmet med dette
   - *fire funksjoner* (som kalles i/brukes av `main`) for å operere direkte på `map`en med å:
     a) **skrive ut *alle* dataene om *alle* de som for øyeblikket er på stedet.** Er det ingen på stedet, kommer det en melding om dette. I motsatt fall skrives selve stedets navn, deretter en *oversikt over alle* de besøkende, og til slutt dette antallet.
     b) **legge inn (om mulig) *en* ny besøkende**. Det spørres først om et aktuelt mobilnr. Finnes denne allerede, kommer det en egen melding. I motsatt fall opprettes en ny besøkende, alle dens data leses inn, og den legges inn i `map`en.
     c) **finne (om mulig) en besøkende**. Er det ingen på stedet, kommer det en melding om dette. I motsatt fall leses et aktuelt mobilnr. Finnes det en besøkende med dette nummeret, så returneres det en peker til aktuelt objekt. I alle andre tilfeller returneres `nullptr`.
     d) **slette (om mulig) en besøkende**. Det letes først etter en ønsket besøkende (bruk funksjonen i pkt.2c). Blir den *ikke* funnet, kommer det en egen melding. I motsatt fall skrives alle dens data *bakerst* på filen 'BESOK.DTA', og den slettes/fjernes *totalt* fra datastrukturen.

   Du skal *ikke* lage destructorer i noen av de to klassene.

3. ***Ett* globalt objekt** av klassen i pkt.2 (der stedets navn sendes med til constructoren)
   Det er dette objektet som `main` bruker for å få tak i de fire funksjonene 2a) - 2d).

Oppgave 3 (20%)

Vi har allerede denne ferdige koden:

```
class Node  {
  private:
     int nummer;    // Andre datamedlemmer er uvesentlig hva er i dette programmet
  public:
     Node(int n)           {  nummer = n;  }
     int  hentID() const   {  return nummer;  }
     void skrivData() const
                    { cout << ' ' << nummer; /* Utskrift andre data */ }
     void lesData()  {  /* Leser inn de andre datamedlemmeme */  }
     void funkA()    {  cout << " funkA"; /*  + annen kode/innmat */ }
     // + Flere funksjoner .....
};
```

1. **Definer en liste bestående av pekere til `Node`'r**.
2. **Lag en funksjon, som mottar en liste som parameter**, og som via dette initierer den originale listen med tyve pekere til objekter, der deres numre er fra 101-120. Objektenes `lesData()` kjøres før innlegging i listen. Noder med `nummer` lik et partall legges inn bakerst i listen, de andre forrest.

Det skal nå videre opereres på listen laget i pkt. nr.1, og initiert ved kall på funksjonen i pkt. nr. 2.

Dere skal *kun* gjøre *en* av de fire blokkene (A, B, C, D) nedenfor, ut i fra sifre i kandidatnummeret deres:

A) ***Siste siffer er partall (0, 2, 4, 6, 8) og nest siste siffer er også partall:***

3. Skriv ut *hele* listens innhold ved å bruke range-based for-løkke
4. Tilkall `funkA()` inni *alle* objektene ved å bruke range-based for-løkke
5. Sjekk og skriv ut «Listen er tom» (dersom den er det), *uten* å bruke `empty()`
6. Skriv ut dataene i det *første* objektet ved bl.a. å bruke en funksjon i `<list>`
7. Skriv ut dataene i det *siste* objektet ved å bruke iterator
8. Bruk funksjon fra `<algorithm>` for å finne (og skrive ut) *antall* noder som har et `nummer` som er et partall (*ikke* bruk `for_each(…)`)
9. Bruk funksjon fra `<algorithm>` for å finne (og skrive ut) det første objektet som har et nummer som er heltallelig delelig ned '9' (*ikke* bruk `for_each(…)`)
10. Slett/fjern deretter dette objektet (fra pkt.9) helt fra listen.
11. Sorter listen ved å bruke en funksjon fra `<list>`
12. Flytt de fem forreste elementene/objektene til bakerst i listen. Dette kan gjøres på flere måter, men vektlegg kort og effektiv kode.
13. Slett/fjern *alt* i listen

B) ***Siste siffer er partall (0, 2, 4, 6, 8) og nest siste siffer er oddetall (1, 3, 5, 7, 9):***

3. Skriv ut *hele* listens innhold ved å bruke `for_each(…)` fra `<algorithm>`
4. Tilkall `funkA()` inni *alle* objektene ved å bruke `for_each(…)` fra `<algorithm>`
5. Sjekk og skriv ut «Listen har innhold» (dersom den har det), *uten* å bruke `empty()`
6. Skriv ut dataene i det *første* objektet ved bl.a. å bruke en funksjon i `<list>`
7. Skriv ut dataene i det *siste* objektet ved å bruke iterator
8. Bruk funksjon fra `<algorithm>` for å finne (og skrive ut) *antall* noder som har et `nummer` som er et oddetall (*ikke* bruk `for_each(…)`)

9. Bruk funksjon fra `<algorithm>` for å finne (og skrive ut) det første objektet som har et nummer som er heltallelig delelig ned '8' (*ikke* bruk `for_each(…)`)
10. Slett/fjern deretter dette objektet (fra pkt.9) helt fra listen.
11. Sorter listen ved å bruke en funksjon fra `<list>`
12. Flytt de fem forreste elementene/objektene til bakerst i listen. Dette kan gjøres på flere måter, men vektlegg kort og effektiv kode.
13. Slett/fjern *alt* i listen

C) ***Siste siffer er oddetall (1, 3, 5, 7, 9) og nest siste siffer er også oddetall:***

3. Skriv ut *hele* listens innhold ved å bruke range-based for-løkke
4. Tilkall `funkA()` inni *alle* objektene ved å bruke range-based for-løkke
5. Sjekk og skriv ut «Listen har innhold» (dersom den har det), *uten* å bruke `empty()`
6. Skriv ut dataene i det *første* objektet ved å bruke iterator
7. Skriv ut dataene i det *siste* objektet ved bl.a. å bruke en funksjon i `<list>`
8. Bruk funksjon fra `<algorithm>` for å finne (og skrive ut) *antall* noder som har et `nummer` som er et partall (*ikke* bruk `for_each(…)`)
9. Bruk funksjon fra `<algorithm>` for å finne (og skrive ut) det første objektet som har et nummer som er heltallelig delelig ned '9' (*ikke* bruk `for_each(…)`)
10. Slett/fjern deretter dette objektet (fra pkt.9) helt fra listen.
11. Sorter listen ved å bruke en funksjon fra `<list>`
12. Flytt de fem forreste elementene/objektene til bakerst i listen. Dette kan gjøres på flere måter, men vektlegg kort og effektiv kode.
13. Slett/fjern *alt* i listen

D) ***Siste siffer er oddetall (1, 3, 5, 7, 9) og nest siste siffer er partall (0, 2, 4, 6, 8):***

3. Skriv ut *hele* listens innhold ved å bruke `for_each(…)` fra `<algorithm>`
4. Tilkall `funkA()` inni *alle* objektene ved å bruke `for_each(…)` fra `<algorithm>`
5. Sjekk og skriv ut «Listen er tom» (dersom den er det), *uten* å bruke `empty()`
6. Skriv ut dataene i det *første* objektet ved å bruke iterator
7. Skriv ut dataene i det *siste* objektet ved bl.a. å bruke en funksjon i `<list>`
8. Bruk funksjon fra `<algorithm>` for å finne (og skrive ut) *antall* noder som har et `nummer` som er et oddetall (*ikke* bruk `for_each(…)`)
9. Bruk funksjon fra `<algorithm>` for å finne (og skrive ut) det første objektet som har et nummer som er heltallelig delelig ned '8' (*ikke* bruk `for_each(…)`)
10. Slett/fjern deretter dette objektet (fra pkt.9) helt fra listen.
11. Sorter listen ved å bruke en funksjon fra `<list>`
12. Flytt de fem forreste elementene/objektene til bakerst i listen. Dette kan gjøres på flere måter, men vektlegg kort og effektiv kode.
13. Slett/fjern *alt* i listen

Annet (klargjørende):

- *Ikke* bruk saker i STL utenfor pensumet, templates eller annet stoff/biblioteker utenfor pensum.
- Gjør dine egne forutsetninger og presiseringer av oppgaven, dersom du skulle finne dette nødvendig. Gjør i så fall klart rede for disse *i starten* av din besvarelse av oppgaven.

Lykke til med *eget, selvstendig og individuelt* arbeid!

FrodeH