Tentamen
EDAF30 Programmering i C++

2015–05–18, 8.00–13.00

Hjälpmedel: En valfri C++-bok. OH-bilderna från föreläsningarna är *inte* tillåtna.

Du ska i dina lösningar visa att du behärskar C++ och även att du kan använda C++ standardklasser. Rena ”C-lösningar” på problem som med fördel kan lösas enligt mera objektorienterade principer kan därför ge poängavdrag, även om de är korrekta.

Uppgifterna ger preliminärt $8 + 12 + 16 + 14 = 50$ poäng. För godkänt krävs preliminärt 25 poäng (betygsgränser 3/25, 4/33, 5/42). Observera att ordningsföljden eller poängantalet för uppgifterna inte nödvändigtvis avspeglar deras svårighet.

---

1. Ordbehandlingsprogram som kan avstava ord använder dels algoritmer som beskriver reglerna för avstavning, dels listor med ord där avstavningarna inte följer reglerna. Den enklaste regeln för avstavning av svenska ord är:

   - Bindestreck får placeras mellan bokstäver på så sätt att direkt efter varje bindestreck följer en konsonant och en vokal.

   Skriv en funktion som implementerar denna regel. Givet ett ord (en sträng innehållande bokstäverna a-z och A-Z) skall funktionen ge en ny sträng med alla möjliga bindestreck instoppade.

   Exempel:

   ```
   algoritmer => al-go-rit-mer
   Ordbehandlingsprogram => Ord-be-hand-lingsp-rog-ram
   katt => katt
   ```

2. Skriv ett program som först frågar efter namnet på en textfil och läser in detta namn från standard input (tangentbordet). Därefter öppnar programmet filen och läser in ett antal ord, separerade med ”whitespace”, från filen och skriver ut orden i bokstavsordning tillsammans med antalet gånger orden förekom i indata. Du skall inte göra skillnad mellan små och stora bokstäver.

   Till exempel skall en fil innehållande följande indata:

   ```
   A dog and a cat and then a dog
   ```

   ge utskriften:

   ```
   a 3
   and 2
   cat 1
   dog 2
   then 1
   ```

   Tips: Funktionen tolower(ch) returnerar tecknet ch omvandlat till liten bokstav om ch är en stor bokstav, ch annars. Du kan förutsätta att användaren matar in ett korrekt filnamn och att filen existerar och har korrekt innehåll.

---

3. En klass `Polynomial` som beskriver polynom i en variabel används på följande sätt:

```
int main() {
    Polynomial p1;
    p1.add_term(-2, 1);
    p1.add_term(3.5, 0);
    p1.add_term(8.2, 4);
    cout << p1 << endl;      // 3.5 - 2x + 8.2x^4
    cout << p1(2.0) << endl; // compute value for x = 2.0, 130.7
    Polynomial p2;
    p2.add_term(10, 7);
    p2.add_term(1, 0);
    p2.add_term(1, 2);
    p2.add_term(-8.2, 4);
    cout << p2 << endl;      // 1 + x^2 - 8.2x^4 + 10x^7
    p1 += p2;
    cout << p1 << endl;      // 4.5 - 2x + x^2 + 10x^7
}
```

   Man kan alltså:

   - skapa ett tomt polynom,
   - lägga till en term med koefficient och gradtal,
   - skriva ut ett polynom,
   - beräkna polynomets värde i en punkt,
   - addera ett polynom till ett annat med +=

   Implementera klassen enligt följande anvisningar:

   - Polynomet *ska* representeras av ett objekt av standardklassen `list` med element som innehåller koefficient och gradtal (skriv en egen klass för att representera dessa eller använd standardklassen `pair`) för de termer vars koefficient inte är noll. Listan *ska* vara sorterad efter växande gradtal.
   - I `add_term` kan du förutsätta att det inte redan finns en term med angivet gradtal.
   - Utskriften behöver inte göras så elegant som i exemplet. Det räcker med att koefficient och gradtal skrivs ut för varje term, med en term per rad.

4. I kursboken (J. Skansholm: C++ direkt) hittar man följande funktion som använder sig av binärsökning för att leta upp positionen (returnerar en pekare till elementet) för ett heltal i ett sorterat fält:

```
int *bin_sok(int sokt, int *forsta, int *sista) {
        if (forsta>sista)
                return 0;
        int *mitten = forsta + (sista-forsta)/2;
        if (sokt<*mitten)
                return bin_sok(sokt,forsta,mitten-1);
        else if (sokt>*mitten)
                return bin_sok(sokt,mitten+1,sista);
        else
                return mitten;
}
```

   a) Skriv om funktionen till en funktionsmall så att man i stället för att bara kunna leta i ett int-fält kan söka efter ett element i ett sorterat fält av godtycklig typ (med element som kan jämföras).

vänd...

b) Antag att vi vill använda din funktionsmall ovan för att söka efter element av klassen MI6Agent enligt nedan. **OBS!** Sökningen ska ske baserat på *numret* för agenten (t.ex. ”007”).

```
class MI6Agent {
public:
    MI6Agent() : agent_number(0), agent_name(0) { }
    void set(const string& nbr, const string& nm) {
        delete agent_number;    // delete 0 is ok in C++ so no null check required...
        delete agent_name;
        agent_number = new string(nbr);
        agent_name = new string(nm);
    }
    string get_number() {
        return *agent_number;
    }
    string get_name() {
        return *agent_name;
    }
private:
    string *agent_number;
    string *agent_name;
};
```

Nedan finns ett litet exempel som visar hur det ska kunna fungera:

```
MI6Agent agent[4];
agent[0].set("001","Donne, Edward");
agent[1].set("004","Papava, Scarlett");
agent[2].set("007","Bond, James");
agent[3].set("009","Moneypenny, Eve");
MI6Agent s;
s.set("007","");
MI6Agent *a = bin_sok(s,agent,agent+3);
if (a!=0) {
    cout << a->get_number() << endl << a->get_name() << endl;
} else {
    cout << "Fanns ej!" << endl;
}
```

Komplettera koden för klassen `MI6Agent` (dvs ändra ej existerande kod - lägg bara till ny kod) så att koden i exemplet ovan fungerar!

c) Ett separat problem med klassen `MI6Agent` i den föregående uppgiften är att man riskerar att råka ut för en minnesläcka när man använder den. Vad innebär en minnesläcka?

d) *Komplettera* koden (ändra ej existerande kod) för klassen `MI6Agent` ovan så att användning av den inte medför en minnesläcka. Var observant så att du inte i och med ändringen introducerar fel som beror på för tidig avallokering av minne.

e) Om du hade fått lov att ändra i den ursprungliga koden för klassen `MI6Agent` skulle det gå att åtgärda problemen med minnesläckor på ett mycket enklare sätt (utan att ändra klassens beteende utåt – det publika gränssnittet). Hur?