Tentamen i TDP004
Objektorienterad Programmering
Praktisk del

Datum:	2008-03-26
Tid:	08-12
Plats:	PC6-PC7 i E-huset.
Jour:	Per-Magnus Olsson, tel 285607 Jourhavande kommer att besöka skrivsalarna varje timme under skrivtiden.
Hjälpmedel:	Teoretisk del: Inga. Praktisk del: En bok om C++.
Betygsättning:	Max antal poäng: 42 med 21 poäng vardera på teori och praktikdel.

Poäng	**Betyg**
36-42	**5**
29-35	**4**
22-28	**3**
0-21	**U**

Anvisningar:	Börja med den teoretiska delen. När du är klar med den lämnar du in den och får den praktiska delen. När du har lämnat in den teoretiska delen kan du inte återvända till den. Skriv svaret på varje uppgift på ett separat blad. Uppgifterna är inte ordnade i svårighetsgrad.

Lycka till!

TDP004 Objektorienterad Programmering
Praktisk del

1. Gör en klass CalculateStatistics med två publika funktioner
`double CalculateMean(const std::list<double>& numbers) const` och
`double CalculateStandardDeviation(const std::list<double>& numbers) const` (7p).

Funktionen `CalculateMean` beräknar medelvärdet $\tilde{x}$ av elementen i listan numbers enligt formeln $\tilde{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$.

Funktionen `CalculateStandardDeviation` beräknar standardavviklelsen σ av elementen i listan enligt formeln $\sigma = \sqrt{\frac{\sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2}{n-1}}$.

2. Du ska göra en del av ett ramverk för att hantera testfall. Gör en funktion enligt följande:

```
bool WriteTestCaseResult(const unsigned int testCaseNumber,
                         const std::string testCaseName,
                         const bool testCaseResult) const;
```

När denna funktion anropas ska en .txt-fil skapas. Denna ska ha namnet `testCaseNumber_testCaseName`. I filen ska numret, namnet och resultatet skrivas på en rad. Inget mer ska göras med filen. Testfallsresultatet ska skrivas som en bool, och detta ska göras med existerande funktionalitet, dvs du ska **inte** skriva en egen if-sats som skriver strängen ”`false`” resp ”`true`” i filen. Ingenting mer ska skrivas i filen. Om funktionen misslyckas med att skapa filen ska funktionen returnera `false`, annars ska funktionen returnera `true`. Om det redan finns en fil med samma namn ska denna skrivas över. (7p)
Exempel:
`WriteTestCaseResult(10, testsearch, true)`

-> I filen 10_testsearch.txt
10 testsearch true

3. Inom musik är det vanligt att man använder olika filter för att uppnå olika effekter. Skapa en filter-basklass med tre subklasser. Dessa ska vara *amplitudskalning*, *bitcrush* samt *tremolo*. Samtliga filterklasser ska ta en konstant av typen `int` som inparameter till konstruktorn, och överlagra funktionen `int Filter(const int input)` vilken utför filtreringen. Beroende på vilken sorts filter det är ska konstanten användas till olika saker. Ett filter med *amplitudskalning* multiplicerar insignalen med konstanten. Ett *bitcrush*-filter gör en division och sedan en multiplikation med konstanten. Ett *tremolo*-filter adderar `sin(n*k)` till insignalen där n är vilket anrop i ordningen till tremolo-filtret som detta anrop är (dvs n =1 för första anropet osv), och k är konstanten. (7p)