

2. Übung des Programmierpraktikums

Abgabetermin: 23. März 2009, 23:59 Uhr

Die Übungen sind grundsätzlich allein zu machen. Gruppenarbeit ist nicht erlaubt. Abzugeben sind jeweils die sinnvoll dokumentierten Programmfiles (Files für Bsp. 5: `bsp_5.cpp`, `bsp_5_fkt.cpp`, `bsp_5_fkt.hpp`) indem Sie diese in Ihr Verzeichnis im globalen Abgabeordner kopieren.

4. Überprüfen Sie, ob die folgenden Rechenregeln der Mathematik auch im Computer gelten, indem Sie rechte und linke Seite nachstehender Gleichungen und die Differenz beider Seiten ausgeben. Die verfügbaren math. Funktionen sind unter `cmath`¹ zu finden und werden über `#include <cmath>` in Ihren Quelltext eingebunden.

$$\begin{aligned}(e^a)^b &= e^{a \cdot b} \\ \log_{10} b &= \frac{\ln b}{\ln 10} \\ \pi/2 - \arccos a &= \arctan \frac{a}{\sqrt{1-a^2}} \\ \sinh a - \cosh a &= -e^{-a}\end{aligned}$$

Testen Sie diese Rechenregeln sowohl für einfach genaue Zahlen (`float`) als auch für doppelt genaue Zahlen (`double`). Geben Sie jeweils die Anzahl von Byte zur Speicherung einer Zahl unter Nutzung von `sizeof`² an und geben Sie die relative Genauigkeit für Ihren Datentyp analog zum Beispiel `Ex390.cpp`³ an. (1 Pkt.)

Die Zahl π wird von den meisten Compilern in obigem Headerfile als `M_PI` bereitgestellt.

Testdaten (a,b): (0.9876, 1.762e - 4), (0.00187, 98),

5. Schreiben Sie ein Programm, welches eine Gleitkommazahl a und eine ganze Zahl n einliest und diese Zahlen in den entsprechenden Datentypen speichert. Geben Sie gleich nach dem Einlesen die reziproken Werte beider Zahlen aus. Berechnen Sie eine neue Gleitkommazahl b , welche auf n Nachkommastellen gerundet ist, d.h., aus 1123.987654 wird 1123.99 bei $n = 2$. (1 Pkt.)

Nutzen Sie evtl. die Funktionen `floor`, `ceil` aus `cmath`⁴ oder die Tatsache, daß bei der Umwandlung einer Gleitkommazahl in eine ganze Zahl die Nachkommastellen wegfallen, siehe Beispiel `Ex320.cpp`⁵ an.

Eingabedaten (a, n): (113.375145, 3), (3.98765, 2)

¹<http://www.cplusplus.com/reference/clibrary/cmath>

²<http://www.uni-graz.at/~haasegu/Lectures/Kurs-C/Examples/DataTypes.cpp>

³<http://www.uni-graz.at/~haasegu/Lectures/Kurs-C/Examples/Ex390.cpp>

⁴<http://www.cplusplus.com/reference/clibrary/cmath>

⁵<http://www.uni-graz.at/~haasegu/Lectures/Kurs-C/Examples/Ex320.cpp>

6. Schreiben Sie ein Programm, welches 3 natürliche Zahlen einliest und davon die Summe, das arithmetische, geometrische und harmonische Mittel berechnet, sowie in Variablen speichert. Die Eingabedaten und Ergebnisse sollen (informativ) ausgegeben werden. Wählen sie die Typen der zu definierenden Variablen nach dem Wertebereich der Ergebnisse aus. (1 Pkt.)

Eingabedaten (i1, i2, i3): (16, 4, 1), (3, 5, 8)

Hinweise:

Schauen Sie sich die folgenden C++-Funktionalitäten an und nutzen Sie diese gegebenenfalls: `sizeof`, `exp`, `log`, `log10`, `acos`, `atan`, `sqrt`, `sinh`, `cosh`, `numeric_limits`, `pow`, `floor`, `ceil`, `static_cast`

Abgabe der Lösungen:

Die Abgabe der Lösungen (*.cpp-Files, *.hpp, ..., **keine** exe-Files) erfolgt ausschließlich durch das Kopieren dieser Files in Ihr (beim erstmal anzulegendes) Verzeichnis `Nachname_Vorname` im globalen Abgabeordner `y: \\pers.ad.uni-graz.at\fs\ou\621\stud_haase\Nachname_Vorname` .

Hinweise hierzu sind unter der LV-Homepage⁶ zu finden.

Die Filenamen **müssen** dem Schema `bsp_nummer`, gefolgt von der Fileextension, entsprechen. Z.B. ist in Beispiel 1 das File `bsp_1.cpp` abzugeben. Andere Filebezeichner zählen als nicht abgegebene Files.

⁶<http://www.uni-graz.at/~haasegu/Lectures/Kurs-C/SS08/index.html>