

# 7. Übung des Programmierpraktikums

Abgabetermin: 2. Juni 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.

---

21. In Aufgabe 19 (Blatt 6) haben wir die Vektorlängen (Feldlängen) nur statisch vereinbaren können. Ändern Sie Ihren Code (alternativ auch die Beispiellösung) entsprechend §6.4<sup>1</sup> des Skriptes so ab, daß nach dem Abfragen der Vektorlänge  $n$  die Vektoren **dynamisch** allokiert werden und dann die Berechnungen aus Aufgabe 19 folgen. Vergessen Sie nicht das explizite Deallokieren des dynamischen Speichers. (1 Pkt.)  
*Eingabedaten* ( $n$ ): (4), (20), (31553)

22. In mathematics, the Fibonacci numbers form a sequence defined recursively by the following formulas:

$$F_n = \begin{cases} 0 & \text{if } n = 0, \\ 1 & \text{if } n = 1, \\ F_{n-1} + F_{n-2} & \text{if } n > 1. \end{cases}$$

- Write a **recursive** function `fibonacci(n)` that computes the  $n$ -th element of the sequence. Show that the ratio of a term with respect to the previous one tends to  $\frac{1 + \sqrt{5}}{2}$ . (1 Pkt.)

*Input data* ( $n$ ): (2), (20), (60)

23. Verändern Sie Ihr Hauptprogramm der vorigen Aufgabe so,
- daß die Variable  $n$  sowohl als **Kommandozeilenparameter**, siehe §7.6<sup>2</sup> des Skriptes
  - als auch über `cin` (wie bislang) übergeben werden kann, falls nicht genügend Kommandozeilenparameter vorhanden sind. (3 Pkt.)
  - Schreiben Sie eine weitere Funktion zur Berechnung der Fibonaccizahlen, welche mittels eines Zählzyklus realisiert ist.
  - Vergleichen Sie die Laufzeiten Ihrer beiden Fibonaccifunktionen (aus Aufg. 22 und 23) mittels der **Zeitmessung** in §11.2<sup>3</sup> des Skriptes.

*Eingabedaten* ( $n$ ): (4), (20), (40), (60)

---

<sup>1</sup><http://www.uni-graz.at/~haasegu/Lectures/Kurs-C/Script/html/node67.html>

<sup>2</sup><http://www.uni-graz.at/~haasegu/Lectures/Kurs-C/Script/html/node79.html>

<sup>3</sup><http://www.uni-graz.at/~haasegu/Lectures/Kurs-C/Script/html/node96.html>

24. Implementieren Sie eine Funktion `riemann`, zum (numerischen) Integrieren einer geg. Funktion  $f(x)$  mit Hilfe der Riemann-Summen (1 Pkt.)

$$\int_a^b f(x)dx = \sum_{j=1}^n f(a + jh) \cdot h$$

wobei das geg. Intervall von  $a$  bis  $b$  in (geg.)  $n$  gleichgroße Intervalle der Länge  $h$  unterteilt wird. Die zu integrierende Funktion ist ebenfalls eine INPUT-Größe der Parameterliste, siehe Beispielfunktion `Bisect3` in §7.8<sup>4</sup> der Vorlesung (auch in `Bisect3.cpp`<sup>5</sup>).

*Testdaten:* `sqrt()` in  $[1, 4]$ , `cos()` in  $[0, \pi/2]$ , `log()` in  $[1, e]$

25. Implementieren Sie eine Funktion `derivative`, die unter Verwendung einer gegebenen Schrittweite  $h$ , die numerische Ableitung der (über)gebenen Funktion  $f(x)$  an einem vorgegebenen Punkt  $x$  berechnet. Verwenden Sie die Formel (1 Pkt.)

$$f'(x) = \frac{f(x + h) - f(x - h)}{2h}$$

*Testdaten:* `sqrt()` in  $x = 4$ , `cos()` in  $x = \pi/6$ , `log()` in  $x = 1$ ,

26. In dieser Aufgabe müssen Sie sich unbedingt **an die gegebenen Filebezeichnungen halten**, die da wären `bsp_26_fkt.hpp` für das Headerfile und `bsp_26_fkt.cpp` für das Sourcefile Ihrer zu schreibenden/geschriebenen Funktionen. Kopieren Sie Ihre Funktionen `derivative` und `riemann` aus den beiden vorigen Aufgaben in Header- und Sourcefile, und testen Sie deren korrekte Nutzung in einem kurzen Hauptprogramm. (1 Pkt.)

\_\_\_\_\_ **Ende für LAK** \_\_\_\_\_

27. Schreiben Sie verbesserte Funktionen `riemann` und `derivative` welche das bestimmte Integral (bzw. den Ableitungswert) mit einer Genauigkeit von  $\epsilon$  zurückgeben. Speichern Sie diese Funktionen ebenfalls in den Files `bsp_26_fkt.hpp`, `bsp_26_fkt.cpp`. Testen Sie deren korrekte Nutzung bei  $\epsilon = 10^{-4}$  mit den Testdaten aus Aufg. 24 und 25 in Ihrem Hauptprogramm. (2 Pkt.)  
Achten Sie insbesondere bei der Differentiation darauf, daß die Schrittweite  $h$  nicht zu klein wird (Maschinen-Epsilon in §3.9<sup>6</sup> bzw. `Reihe.cpp`<sup>7</sup>).

---

<sup>4</sup><http://www.uni-graz.at/~haasegu/Lectures/Kurs-C/Script/html/node81.html>

<sup>5</sup><http://www.uni-graz.at/~haasegu/Lectures/Kurs-C/Examples/Bisect3.cpp>

<sup>6</sup><http://www.uni-graz.at/~haasegu/Lectures/Kurs-C/Script/html/node45.html>

<sup>7</sup><http://www.uni-graz.at/~haasegu/Lectures/Kurs-C/Examples/Reihe.cpp>

28. Die Bestimmung der Nullstelle einer Funktion  $f(x)$  ist sehr effizient berechenbar mit der Newton-Iteration<sup>8</sup>

$$x_{n+1} := x_n - f(x_n)/f'(x_n)$$

für einen geeigneten Startwert  $x_0 \in [a, b]$  (die math. Voraussetzung, daß die Funktion  $x_n - f(x_n)/f'(x_n)$  eine kontraktive Abbildung ist, muß in Ihrem Programm nicht getestet werden). (4 Pkt.)

Nutzen Sie Ihre Funktionen aus den Aufgaben 25 und 26 und erweitern Sie die Files *bsp\_26\_fkt.hpp*, *bsp\_26\_fkt.cpp* um Ihre Funktion `newton_diff_approx()`. Testen Sie Ihren Nullstellenloeser (und überprüfen Sie die Lösung) im Hauptprogramm mit den folgenden Funktionen:

*Testdaten:*  $f(x) := \sin(x) - x/2$   
 $g(x) := -(x - 1.234567)(x + 0.987654)$   
 $h(x) := 3.0 - e^x$   
 $z(x) := \cos(x) - e^{x \ln|x|}$

generelle Hinweise:

Die Befehle `break`, `continue`, `goto` sind **nicht erlaubt**.

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-Homepage<sup>9</sup> 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.

---

<sup>8</sup>[http://de.wikipedia.org/wiki/Newton\\_Iteration](http://de.wikipedia.org/wiki/Newton_Iteration)

<sup>9</sup><http://www.uni-graz.at/~haasegu/Lectures/Kurs-C/SS09/index.html>