

### Kreisfläche

In diesem Praktikum werden erste Erfahrungen mit der Berechnung von Ausdrücken gesammelt.

Unter Nutzung der Gleitpunktarithmetik soll der Kreisflächeninhalt zu einem einzugebenden Durchmesser berechnet werden. Die Eingabe des Durchmessers erfolge in der Art und Weise, die im Praktikum 3 (Datentypen/Variable) vermittelt wurde. Um eine Zeichenkette in einen Gleitpunktwert vom Typ float oder Double zu konvertieren, steht die Funktion **double atof(const char\*)**; zur Verfügung. Für genauere Informationen zu dieser Funktion geben Sie in der Konsole **man atof** ein oder recherchieren Sie bei [www.cplusplus.com](http://www.cplusplus.com). Für die Zahl PI können Sie **M\_PI** verwenden, müssen aber **math.h** per Includeanweisung einschließen.

Schreiben Sie ein C-Programm, das mit folgendem Dialog den Kreisflächeninhalt berechnet.

Ein-/Ausgabe:

Berechnung der Kreisfläche

Eingabe des Durchmessers in mm: 10

Flächeninhalt in cm<sup>2</sup>: 0.7850

Dieser Teil des Praktikums sollte keine größeren Schwierigkeiten bereiten.

In vielen Anwendungen ist es jedoch nötig, auf Gleitkommaarithmetik zu verzichten. Dies kann der Fall sein, wenn in einem Prozessor keine solche Gleitpunktarithmetik existiert, wenn Ganzzahlarithmetik schnellere Ergebnisse erzielt oder wenn aus Gründen der Genauigkeit ganzzahlig gerechnet werden muss.

Für unser Beispiel ist dabei zu beachten:

An Stelle von PI wird mit 314 gerechnet. Natürlich müssen wir dann am Ende wieder durch 100 teilen.

Gleichrangige Multiplikationen und Divisionen müssen so notiert werden, dass zunächst multipliziert und am Ende dividiert wird. Dabei ist zu beachten, dass der größtmöglich darstellbare Zahlenwert nicht überschritten wird.

Vor- und Nachkommastellen werden getrennt berechnet und nacheinander durch einen Dezimaltrenner ('.') getrennt ausgegeben.

Um ganze Zahlen in fester Länge mit führenden Nullen auszugeben, verwendet man in printf das Formatsteuerzeichen **%0nd**, wobei n die Anzahl der auszugebenden Stellen ist

z.B. `printf("%04d", i);`. Geauere Informationen findet man unter `man format` oder unter [www.cplusplus.com](http://www.cplusplus.com).

Das Ergebnis sollte sich optisch von dem im ersten Teil des Praktikums nicht unterscheiden.

---

#### Anweisungen Teil 1:

Schreiben Sie zwei C-Programme, die jeweils ein einzugebendes Zeichen untersuchen. Es soll festgestellt werden, ob es sich um einen Großbuchstaben, einen Kleinbuchstaben, eine Zahl oder ein Sonderzeichen handelt.

Die Eingabe soll wieder mittels `fgets(buf, 128, stdin);` erfolgen.

Realisieren Sie je ein Programm unter Verwendung von

- verschachtelten if-Anweisungen
- einer switch Anweisung

Das Programm soll Eingabe und Auswertung des eingegebenen Zeichens in einer Schleife wiederholen, bis die Zeichenfolge "qq" eingegeben wird.

Hinweise zur Lösung:

- Das auszuwertende Zeichen befindet sich in `buf[0]`. Um das Programm zu verlassen, testen Sie bitte `buf[0]` und `buf[1]` auf den Wert 'q'.
- Bei der Variante switch/case benötigen Sie für jedes zu bewertende Zeichen ein Caselabel. Diese können auch hintereinander stehen.

Eine [Ascii-Code-Tabelle](#) finden Sie hier.

---

#### Anweisungen Teil 2:

#### Praktikum Programmiersprache C - Numerische Berechnung des Sinus

Aus der Vorlesung kennen Sie das Beispiel zur Berechnung von  $e^x$ . Der in der Vorlesung entwickelte Algorithmus ist nun auf die unten angegebene Reihenentwicklung zur Berechnung von  $\sin(x)$  anzuwenden. Zu beachten ist, dass nur jeweils jedes 2. Glied zu berechnen ist und dass das Vorzeichen der Summanden alterniert.

## Numerische Berechnung von Funktionswerten der Funktion sin(x)

Schreiben Sie ein Programm zur näherungsweisen Berechnung von sin(x). Sin(x) kann durch summieren der ersten n Glieder einer Reihe, die folgendermaßen gebildet, berechnet werden.

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{(2n+1)!}$$

Die Berechnung soll auf 4 Stellen nach dem Komma genau sein, d.h., die Berechnung kann beendet werden, wenn der berechnete Summand so klein geworden ist, dass das auf die geforderte Stelle gerundete Ergebnis nicht mehr beeinflusst wird.

Beginnen Sie mit dem Summanden x und bilden Sie dann, wie im Vorlesungsbeispiel den nächsten Summanden aus dem vorhergehenden durch Multiplikation und Division.

Berechnen Sie nicht Zähler und Nenner getrennt, weil die Zahlen sehr schnell zu groß werden.

Durch Aufruf der Standardfunktion sin(x) kann das Ergebnis überprüft werden.

(#include < math.h> erforderlich).

Das alternierende Vorzeichen kann durch zyklisches Multiplizieren mit -1 realisiert werden. In diesem Fall muss von dem berechneten Glied beim Vergleich für den Abbruch der absolute Betrag gebildet werden. Dies geschieht durch Aufruf von **double**

**fabs(double) ;**

(#include < math.h> erforderlich).

Auf Unix Platttformen ist die Bibliothek libm einzubinden. Dies kann über die Angabe der Option -lm erfolgen. (gcc sinusBeispiel.c -lm)

---

A. Beck