3.3 C-Datentypen/Gleitkommazahlen und Mathe-Lib

3.3.1 Ein Problem aus der Elektrotechnik

Eine Maschine hat die Wirkleistung $P_W = 4 \text{ kW}$ und die Scheinleistung $P_S = 5 \text{ kVA}$. Gesucht ist der Phasenwinkel φ :

$$\cos \varphi = \frac{P_W}{P_S} = 0,8$$

Damit ist

$$\varphi = \arccos \frac{P_W}{P_S} = \arccos 0,8$$

Der Zahlenwert soll nun natürlich durch ein C-Programm bestimmt werden können.

3.3.2 Mathe-Bibliothek

Für solche Aufgabenstellungen aus der Technik benutzt man meistens Gleitkommazahlen. Gleitkommazahlen entsprechen wegen ihrer weitgehend gleichbleibenden relativen Genauigkeit den meisten Anforderungen in der Technik:

```
1 double Pw=4.0e3, Ps=5.0e3, phi;
```

Die Sprache C bietet nun für Gleitkommazahlen entsprechende mathematische Funktionen an, deren Prototypen über die Header-Datei `<math.h>` eingebunden werden können. Man spricht von der Mathematik-Bibliothek der Sprache C:

```
1 double sin(double x); /* Sinus */
2 double cos(double x); /* Cosinus */
3 double tan(double x); /* Tangens */
4 double asin(double x); /* Arcussinus */
5 double acos(double x); /* Arcuscosinus */
6 double atan(double x); /* Arcustangens */
7 double atan2(double y, double x); /* Arcustangens von y/x */
8 double sinh(double x); /* Sinus Hyperbolicus */
9 double cosh(double x); /* Cosinus Hyperbolicus */
10 double tanh(double x); /* Tangens Hyperbolicus */
11 double exp(double x); /* e hoch x */
12 double log(double x); /* ln x fuer x>0 */
13 double log10(double x); /* lg x fuer x>0 */
14 double pow(double x, double y); /* x hoch y, falls Erg. reell */
15 double sqrt(double x); /* Quadratwurzel von x, falls x>=0 */
16 double ceil(double x); /* kleinste Ganzzahl >= x */
17 double floor(double x); /* groesste Ganzzahl <= x */
18 double fabs(double x); /* absoluter Betrag von x */
19 double ldexp(double x, int n); /* x mal (2 hoch n) */
20 double frexp(double x, int *exp); /* Mantisse von x */
21 double modf(double x, double *ip); /* Nachkommaanteil von x */
22 double fmod(double x, double y); /* Divisionsrest von x/y */
```

Alle Winkelfunktionen beziehen sich übrigens auf das Bogenmaß! Die zweiten Parameter von `frexp()` und `modf()` haben einen Zeigerdatentyp (siehe später) und dienen dazu, jeweils einen weiteren Ergebnisanteil zu speichern.

Die bisher genannten Funktionen benutzen alle den Datentyp `double`. Seit C99 gibt es dieselben Funktionen auch für die Datentypen `float` und `long double`. Die entsprechenden Funktionen heißen dann nicht `sin()`, sondern `sinf()` und `sinl()`:

```

1  float      sinf(float x);
2  double     sin (double x);
3  long double sinl(long double x);

```

3.3.3 Nutzung

Erstens müssen die Prototypen der Mathe-Bibliothek über die Header-Datei `<math.h>` eingebunden werden:

```

1 #include <math.h>

```

Zweitens ist es beim GNU-C-Compiler so, dass die Mathe-Bibliothek aus Platzgründen standardmäßig nicht mit eingebunden wird. Deshalb muss der Compileraufruf erweitert werden um die Option `-lm`:

```

Terminal
schueler@debian964:~$ gcc meinprog.c -lm

```

3.3.4 Lösung

Eine mögliche Lösung zeigt `src/phasenwinkel.c`:

```

1 #include <math.h>
2 #include <stdio.h>
3 int main(void)
4 {
5     double pw=0.0, ps=0.0, phi;
6     printf("Eingabe_Wirkleistung_in_W...: ");
7     scanf("%lf", &pw); while(getchar()!='\n'){}
8     printf("Eingabe_Scheinleistung_in_VA: ");
9     scanf("%lf", &ps); while(getchar()!='\n'){}
10
11     phi=atan(pw/ps); /* Ergebnis im Bogenmass! */
12
13     printf("Phasenwinkel_in_Grad.....: %.1f\n",
14           phi*360.0/M_PI); /* M_PI ist 3,14159.... */
15     return 0;
16 }

```

3.3.5 Fehlerbehandlung

Bei Überlauf liefern die Funktionen aus `<math.h>` den Wert der Konstanten `HUGE_VAL`. Fehler erkennt man allgemein daran, dass die globale Variable `errno` aus `<errno.h>` einen der Werte `EDOM` (Argumentfehler) oder `ERANGE` (Ergebnisfehler) aufweist.

3.3.6 Grenzwerte in `float.h`

In der Headerdatei `<float.h>` befinden sich etliche Konstanten, mit denen man bestimmte Grenzwerte für Gleitkommazahlen auf dem aktuellen System erhält. Hier sind ein paar davon für den Datentyp `double`:

- `DBL_MAX` – größte darstellbare Zahl
- `DBL_MIN` – kleinste normalisierbare Zahl¹

¹Bei noch kleineren Zahlen sinkt die relative Genauigkeit

- `DBL_EPSILON` – kleinste Zahl x , bei der $1.0 + x \neq 1$ gilt