6. Der C-Präprozessor

Bevor ein C-Quelltext compiliert wird, können noch formale Ersetzungen vorgenommen werden. Dies geschieht mit dem sog. "Präprozessor". Präprozessoranweisungen beginnen mit einem #. Da sie rein zeilenorientiert sind, stehen sie immer am Zeilenanfang, bzw. als erstes und einziges in einer Zeile und werden nicht durch einen Strichpunkt terminiert. Zunächst sind folgende Präprozessorbefehle wichtig:

- #include zum Einbinden von Dateien:
  - #include <Standardbibliotheksdatei> setzt an diese Stelle die genannte Datei aus der Standardbibliothek (d. h. der Suchpfad ist implementationsabhängig).
  - #include "Meinedatei" dient zum Einbinden eigener Dateien.

  #include Anweisungen können geschachtelt werden, d.h. es können Dateien eingebunden werden, die ihrerseits #include Anweisungen enthalten.

- #define zum Defininieren eines Makros:
  - Einfache Ersetzung; Beispiel:

```
#define MAX 1000
```

bewirkt, dass ab dieser Anweisung alle `MAX` mit `1000` ersetzt werden. Konstanten sollte man niemals explizit in den Quellcode schreiben, sondern immer solchermaßen formal definieren. Es ist Konvention (wenn auch syntaktisch nicht zwingend), dass so definierte Konstanten großgeschrieben werden.
Makroersetzung (ähnlich zu Funktionen):

```
#define CMULT(x)  (3.8*(x))
#define SQ(x)     ((x)*(x))
#define MAX(a,b)  ((a) > (b) ? (a) : (b))
```

Vorteil: Makroersetzungen funktionieren für alle Typen.
Nachteil: Die Wahrscheinlichkeit von Seiteneffekten ist hoch; so ist z. B. die exzessive Klammerung in den obigen Beispielen absolut notwendig, würde die Definition ohne Klammern erfolgen, z. B.SQ(x) x * x, dann ergäbe der Aufruf SQ(2 + 5) nicht etwa 7 * 7 = 49, sondern als Textersetzung 2 + 5 * 2 + 5" = 2 + 10 + 5 = 17.
Die #define Anweisung lässt sich rückgängig machen mit dem #undef Befehl.

```
#undef MAX
```

- #if, #else, #endif: Bedingte Kompilierung.

```
#if  MAX < 1000
  #define  MIN  100
#else
```

```
  #define  MIN  200
#endif
```

Allgemein wertet #if einen konstanten Integerausdruck aus; statt < kann irgendein Vergleichsoperator stehen. Weiterhin lässt sich mit #ifdef name verzweigen, wenn dieses Marke definiert, bzw. mit #ifndef name nicht definiert ist.

```
#ifdef  DEBUG
  printf("a hat hier den Wert:  %f\n", a);
#endif
```

Eigene Präprozessor-Fehlermeldungen lassen sich mit #error ausgeben, was außerdem den Compilierungsvorgang abbricht.

```
#ifndef  AFLAG
   #error  "AFLAG" nicht definiert!
#endif
```

Einbinden von Headerdateien

Headerdateien dienen der Modularisierung von Programmen. Dadurch wird die Erweiterbarkeit und die Wartbarkeit von Programmen erhöht.

Eine Headerdatei beinhaltet Deklarationen und andere Bestandteile des Quelltextes. Quelltext, der sich in einer Header-Datei befindet, ist im Allgemeinen zur Verwendung in mehreren Teilen des Programmes vorgesehen. Es werden dadurch Funktionen und Variablen im Porgramm bekannt gemacht.

Den Inhalt von Header-Dateien bilden üblicherweise

- die Vereinbarung von Makros und symbolischen Konstanten,
- die Beschreibung der Aufrufschnittstelle von externen Funktionen,
- die Deklaration von nachnutzbaren Datentypen.

Um das mehrfache einbinden von Headerdateien und dadurch Linkerfehler durch mehrfach Deklarationen zu vermeiden, muss sichergestellt werden, dass eine Headerdatei nur einmal eingebunden wird. Dies geschieht mit dem unten dargestellten Schema.

Headerdatei

```
#ifndef NAME_DER_HEADERDATEI_H
#define NAME_DER_HEADERDATEI_H

Code der Headerdatei

#endif
```

Sourcecodedatei

```
#ifndef MY_CONIO_H
#include "my_conio.h"
#endif

Code der Sourccodedatei
```