1.3 Handhabung eines Programmiersystems

Der Turbo-Pascal-Editor

Man muss nichts vom Innenleben eines Computers verstehen, um mit ihm umgehen zu können. Wir wollen ihn als "black box" verwenden, also lediglich seine Fähigkeiten nutzen, ohne sein physikalisches Innenleben zu verstehen.

Nach dem Einschalten des Computers führt der PC einen Selbsttest durch, bei dem alle Systemkomponenten (Prozessor, Speicher, Schnittstellen) überprüft werden. Dies geschieht durch das in einem ROM-Speicher abgelegte sog. BIOS-Programm. Dann wird das Betriebssystem vom Netzwerkserver oder mit Hilfe einer sog. Systemdiskette gestartet und anschließend das Turbo-Pascal-Programmiersystem aufgerufen. In dem an der Schule installierten Netzwerk geschieht dies durch Anmeldung im Netz und anschließende Auswahl von Turbo-Pascal. Dann steht dessen Editor (im wesentlichen ein Textverarbeitungsprogramm) zur Verfügung, mit dem das zu erstellende Programm eingetippt werden kann. Ein fertiges Programm (Quellcode) muss erst mit einem weiteren Programm, dem Compiler, in einen maschinenverständlichen Code übersetzt (compiliert) werden. Dann kann es mit dem Kommando RUN gestartet werden. Mit den Befehlen LOAD und SAVE werden Programme geladen bzw. auf einem externen Medium gespeichert.

Anmerkungen zum Editor bzw. zu Turbo-Pascal:

1. Die einzelnen Menüpunkte werden angewählt, indem man bei gedrückter ALT-Taste den farblich abgesetzten Buchstaben des gewünschten Befehls in der Menüzeile drückt. Dann erscheint ein Pull-Down-Menü mit weiteren Befehlen.
2. Die Befehle sind in einzelnen logisch zusammenhängenden Gruppen zusammengefasst.
3. Die Übersetzung des Programms in die Maschinensprache kann durch einen Interpreter, z. B. BASIC, oder einen Compiler, z. B. PASCAL, in Maschinensprache oder einen maschinenspracheähnlichen Objektcode übersetzt und dann ausgeführt. Ein Interpreter liest den Programmtext (Quellcode) Anweisung für Anweisung erst beim Programmlauf und führt ihn sofort aus (Vergleich: Simultanübersetzer). Ein Compiler übersetzt den Quellcode als Ganzes nach erfolgreicher Fehlerbeseitigung in den Objektcode oder direkt in Maschinensprache und führt ihn dann aus (Vergleich: Übersetzen eines Buches und lesen der Übersetzung).

Grundelemente von Turbo-Pascal

Anmerkung zur Schreibweise: Sog. reservierte Wörter (Bezeichner des Sprachmittels) erscheinen nachfolgend im Fettdruck, Syntax und Anmerkungen sowie Beispiel in Normaldruck.

Jedes Pascal-Programm wird in drei große Abschnitte eingeteilt:

1. Programmkopf
2. Deklarationsteil (mit Prozeduren/Funktionen)
3. Hauptprogramm

Zu 1.: Der Programmkopf besteht aus dem reservierten Wort **PROGRAM** und dem Programmnamen (ohne Umlaute und ähnliche Sonderzeichen).

Zu 2.: Hier werden **Units** (Bibliotheksteile) aufgerufen und Konstanten, Datentypen und Variablen vereinbart, die im Programm vorkommen und gebraucht werden. Weiterhin folgen alle Prozeduren (Unterprogramme, die vom Hauptprogramm aufgerufen und dann abgearbeitet werden) und Funktionen des Programms. Prozeduren haben die gleiche Form wie ein Hauptprogramm selbst, d. h. sie bestehen aus dem Prozedurkopf (**PROCEDURE** ...) und dem Prozedurrumpf.

Zu 3.: Das Hauptprogramm fängt mit **BEGIN** an und hört mit **END** auf. Es besteht jeweils aus einzelnen Anweisungen und Blöcken von Anweisungen, die jeweils durch **BEGIN** und **END** zusammengehalten werden.

Zu den Turbo-Pascal-Befehlen sind folgende grundsätzlichen Anmerkungen zu machen:

1. Ein Kommentar zum Programm wird in geschweiften Klammern {Kommentar} geschrieben. Da viele Rechner diese Zeichen nicht darstellen können, werden oft die Ersatzzeichen (* und *) verwendet. (Beispiel: (* Dies ist ein Kommentar.*))
2. Alle Pascal-Befehle werden durch ein Semikolon (;) voneinander getrennt.
3. Pascal-Befehle werden immer in der Reihenfolge abgearbeitet, in der sie auftreten.
4. Die Anweisungen des Programms werden zwischen **BEGIN** und **END** geschachtelt.
5. Nach **BEGIN** steht kein Semikolon!
6. Vor **END** kann ein Semikolon stehen, sollte jedoch nicht. (Nur ein Semikolon erzeugt nämlich eine leere Anweisung.)
7. Nach **END** steht ein Punkt, wenn es das Ende des Hauptprogramms darstellt, sonst ein Semikolon.
8. Groß- und Kleinschreibung von Namen für Befehle, reservierte Wörter, Operatoren usw. werden von Turbo-Pascal gleich behandelt.

Ein Programm mit jeweils einer Prozedur und einer Funktion hat das folgende grundsätzliche Aussehen:

```
PROGRAM demo1;

USES      unit1, unit2, ...;
VAR       ...;
CONST     ...;
TYPE      ...;

PROCEDURE prodedurname1;
      VAR       ...;
      CONST     ...;
      BEGIN
            ...
      END;

FUNCTION funktionsname1 (werteparameter) : ergebnistyp;
      VAR       ...;
      CONST     ...;
      BEGIN
            ...
      END;

BEGIN {Hier erst beginnt das Hauptprogramm}
            ...
END.
```

Nach diesen grundsätzlichen Hinweisen soll endlich das erste Programm entstehen und laufen.

Beispiel: Berechne Umfang U und Inhalt A eines Rechtecks mit Länge l und Breite b.

Struktogramm:

Lösche den Bildschirm.
Gib eine Zahl ein und nenne sie laenge.
Gib eine Zahl ein und nenne sie breite.
Berechne den Umfang durch U = 2 · (laenge + breite).
Schreibe den Wert von U auf den Bildschirm.
Schreibe den Wert des durch A = laenge · breite bestimmten Flächeninhalts A auf den Bildschirm.
Beende die Arbeit.

In Turbo-Pascal wird dieses Programm so codiert:

```
program programm1;

uses crt;
var  laenge, breite, umfang: real;

begin
     clrscr; {löscht den Bildschirm}
     write('Länge: ');
     readln(laenge);
     write('Breite: ');
     readln(breite);
     umfang := 2 * (laenge + breite);
     writeln('Umfang: ', umfang:5:2);
     writeln('Fläche : ', laenge * breite:5:2);
     readln
end.
```

Anmerkungen:

1. **CRT** ist eine Unit, in der unter anderem die Bildschirmausgabe kontrolliert wird. In ihr ist z. B. die Prozedur **CLRSCR** zum Löschen des Bildschirms abgelegt.
2. Mit der Prozedur **READ** wird auf eine Tastatureingabe gewartet und diese der angegebenen Variablen zugewiesen. Die Eingabe wird mit Druck auf die "Return"-Taste abgeschlossen. **READLN** bewirkt zusätzlich einen Zeilenvorschub (CR, Carriage Return).
   Syntax: **READLN**(variable1, variable2, ...);
   Beispiel: READ(r);

3. Die Prozedur **WRITE** schreibt Texte, Werte von Variablen, Daten usw. in ein beliebig wählbares Ausgabegerät (Bildschirm, Drucker, Datei usw.). Text muss stets durch Hochkomma angeführt werden. Zahlen können formatiert ausgegeben werden; das Format (Stellenzahl, Nachkommalänge) steht hinter der Variablen, von dieser und untereinander durch Doppelpunkte getrennt. **WRITELN** bewirkt einen zusätzlichen Zeilenvorschub.
   Syntax: **WRITE**(gerät, 'Text', variable:Stellen:Nachkomma);
   Beispiel: WRITELN(lst, 'Alles klar!', 2*r:5:2);
4. Das letzte **READLN** bewirkt, dass das Programm nicht sofort nach dem Programmlauf wieder in den Editor zurückspringt, sondern auf eine erneute Eingabe (hier genügt ein Druck auf die "Return"-Taste) wartet.
5. Im vorliegenden Programm werden bereits konstante und variable Daten verarbeitet. Sie werden in den nächsten Kapiteln noch genauer besprochen.
6. Wertzuweisungen können als selbständige Anweisungen durch **:=** erfolgen, aber auch innerhalb einer Ausgabeanweisung durch Angabe der entsprechenden Formel erfolgen.
   Syntax: variable **:=** mathematischer oder logischer Zusammenhang.