Aufgaben zum Einarbeiten in die Turtle

Aufgabe 1: Lies die erste halbe Seite der Anleitung durch, starte dann Turbo Pascal und tippe das Programm im zweiten Kasten (`PROGRAM Ein_n`) ein. Beachte dabei:

- F3 Datei Öffnen
- F2 Datei speichern
- Alt-F-A Datei unter einem anderen Namen speichern
- F9 Prüfen, ob das Programm keine Tippfehler enthält. Wenn es einen Fehler gibt, dann wird er in einem roten Balken beschrieben, und der Cursor steht an (oder genau hinter) der Stelle des Fehlers.
- STRG-F9 Programm laufen lassen
- F7 Programm Schrittweise laufen lassen. Mit Alt-F5 kann man sich nach jedem Schritt das bis dahin entstandene Bild ansehen.
- Alt-X Programm Beenden
- STRG-Y Die aktuelle Zeile löschen

Diese Befehle (außer Zeile löschen) kann man auch alle über die Menus anwählen, es ist aber praktischer, wenn man sich diese Tasten angewöhnt. **Wichtig: Bevor Du das Programm laufen lässt, solltest Du es abspeichern, sonst könnte bei einem Programmfehler Deine ganze Arbeit verlorengehen.**
Das Menu Edit entspricht dem Windowsmenu Bearbeiten: Copy, Paste (Einfügen), Cut. Diese Funktionen kann man auch benutzen, um Programmteile vn einem Programmfenster in ein anderes zu übertragen. **Mit ALT-2 wechselt man z.B. in das Fenster Nr. 2**

Wenn Du das Programm abgetippt hast, speichere es und lasse es laufen. Versuche anhand der Beschreibung in der Anleitung zu verstehen, wie das Programm funktioniert. Versuche, das N größer zu machen, indem Du das Programm an geeigneten Stellen abänderst.

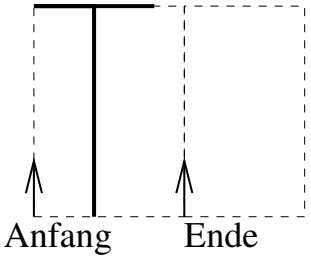
Aufgabe 2: Schreibe ein neues Programm, welches die unabänderlichen Turtle-Zeilen enthält (siehe erster Kasten), und lasse die Turtle in diesem Programm eine beliebige Figur zeichnen. Du kannst zwischendurch immer wieder das Programm starten, umzu sehen, was Du bis dahin geschafft hast.Experimentiere auch mit penup und pendown.

Aufgabe 3: Schreibe ein Programm, welches das Wort ELLE auf den Bildschirm schreibt.

Prozeduren

Aufgabe 4: Lies den unteren Teil von Seite 2 der Anleitung, und verwende die Information, um das Programm mit dem Wort ELLE unter Verwendung von Prozeduren neu und besser zu schreiben.

Wenn man Prozeduren für Buchstaben schreibt, sollten sich diese beliebig kombinieren lassen. Dazu ist es notwendig, dass am Anfang und am Ende jeder Buchstabenprozedur die Turtle gleich steht. Dazu denkt man sich am besten einen Kasten (gestrichelt), in den der Buchstabe passt, und der rechts noch etwas mehr Platz als Lücke zwischen den Buchstaben enthält.



Rechts davonn kann dann sofort der nächste Buchstabe kommen, das Ende der ersten Prozedur ist zugleich die Ausgangsposition der folgenden Prozedur. Sieh Dir am Kasten auf S2 der Anleitung an, wie das dort realisiert ist.

Aufgabe 5: Schreibe nun ein neues Programm, mit Prozeduren zu den Buchstaben E, L, F, H, I, T, M, N, A, K, W, Z. Die Buchstaben sollen gleich hoch sein. Im Hauptprogramm kannst Du aus diesen Buchstaben Wörter zusammensetzen.

Schleifen

Nun möchte man auch Buchstaben mit Rundungen zeichnen lassen. Rundungen erhält man, indem man ein Vieleck zeichnet, also sehr oft hintereinander beispielsweise

```
forwd(2);
turnright(10);
```

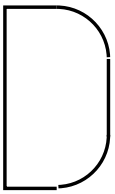
ausführen lässt. Es wäre unpraktisch, diese beiden Zeilen 36 mal ins Programm zu schreiben. Für die mehrmalige Wiederholung einer Befehlgruppe gibt es das Mittel der Zählschleife.

```
for i := 1 to 36 do
  begin
    forwd(2);
    turnright(10);
  end;
```

Man braucht dazu eine Variable, hier **i**, die die Anzahl der Wiederholungen zählt. Durch `i:=1` wird sie erst einmal auf 1 gesetzt. Dann wird der Schleifenkörper, der Teil zwischen `begin`... end *ausgeführt. Anschließend wird der Wert von **i** um 1 erhöht, und wenn er noch nicht größer als 36 ist, wird der Schleifenkörper nochmal durchlaufen. Das geschieht so oft, bis der Wert von **i** größer als 36 ist. (Weitere Informationen zu Schleifen siehe die kurze Anleitung, S 4).
Dem Programm muss man übrigens mitteilen, dass man eine Variable braucht. Das tut man entweder vor dem `begin` des Hauptprogrammes, wenn die Schleife im Hauptprogramm ist, oder vor dem `begin` der Prozedur.

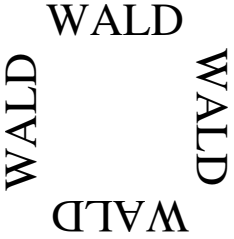
Aufgabe 6: Experimentiere in einem neuen Programm mit dem Mittel der Schleife: Lass das Programm verschieden große Kreise und Halb- und Viertelkreise zeichnen. Überlege, wie der Drehwinkel und die Anzahl der Wiederholungen aufeinander abgestimmt sein müssen.

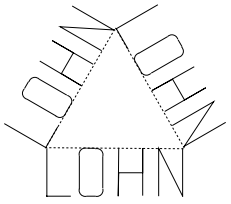
Aufgabe 7: Nun zu den Buchstaben mit Rundungen. Ein Problem ist, etwa beim Buchstaben **D** den Halbkreis so dem senkrechten Strich anzupassen.Ein Trick ist, wie in der nebenstehenden Skizze, den Bogen mit zwei Viertelkreisen und dazwischen liegenden geraden Stücken zu zeichnen. An den geraden Stellen kann man den Bogen beliebig formen.

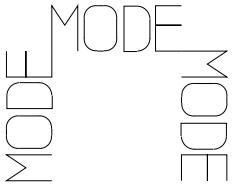


Schreibe nun Prozeduren für in der Größe passende Buchstaben O, D, P, B, ...

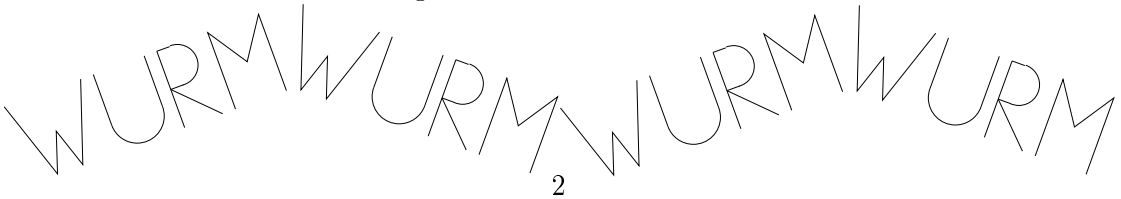
Aufgabe 8: Zum krönenden Abschluss der Buchstaben schreibe nun Programme, welche die folgenden Figuren zeichnen. Natürlich wird das Wort in einer Prozedur gemalt, welche ihrerseits die Buchstabenprozeduren aufruft.







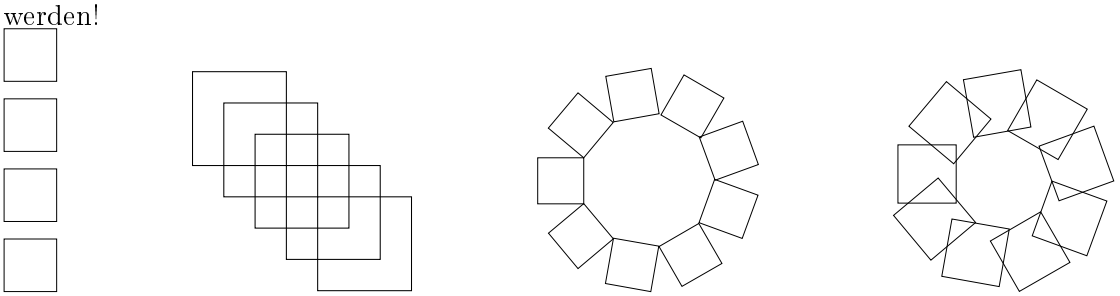
und als besonders schwere Übung:



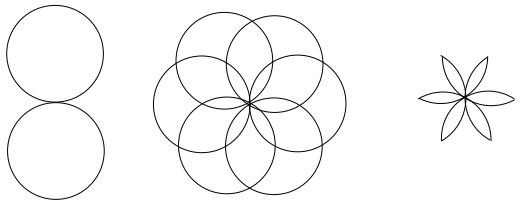
Weitere Übungen zu Schleifen und Prozeduren

Nun verlassen wir die Buchstaben. In den folgenden Aufgaben soll der weitere Umgang mit Prozeduren und Schleifen geübt werden:

Aufgabe 9: Schreibe Programme, welche die folgenden Gebilde zeichnen. Alle Aufgaben beruhen auf einer Prozedur, welche ein Quadrat zeichnen. Es müssen Schleifen verwendet werden!

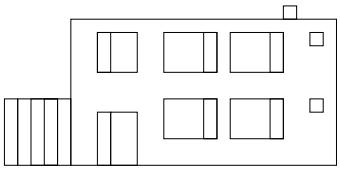


Aufgabe 10: Und nun das gleiche mit Kreisen. Besonders anspruchsvoll ist Aufgabe c). Aus was für Kreisbogenteilen besteht die Blume, - und wie groß ist der Drehwinkel an den Spitzen?



Prozeduren mit Parametern

Das Bungalow besteht nur aus Rechtecken, und diese unterscheiden sich nur in der Länge der Seiten. Man würde sich jetzt wünschen, dass man einer Rechteckprozedur die Seitenlängen mitteilen kann. Dazu wird der Prozedurkopf um die "Parameterliste" verlängert:



```
Procedure Rechteck(h,b: integer);
  begin
    forwd(h);
    turnright(90);
    forwd(b);
    turnright(90);
    forwd(h);
    turnright(90);
    forwd(b);
    turnright(90);
  end;
```

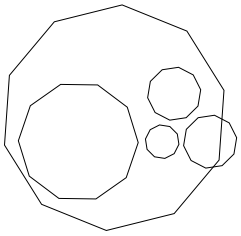
Die Klammer bedeutet, dass diese Prozedur, wenn sie aufgerufen wird, zwei ganze Zahlen (Integer) mitgeteilt bekommt, nämlich **h** und **b**. Ruft man nun im Hauptprogramm auf:

```
rechteck(3,9);
rechteck(7,4);
```

so wird zuerst ein Rechteck gezeichnet, bei dem $h = 3$ und $b = 9$ ist, und danach eines mit $h = 7$ und $b = 4$. Die Zuordnung richtet sich alleine nach der Reihenfolge: Die erste Zahl beim Aufruf wird an den ersten Parameter in der Klammer geschickt, usw.

Aufgabe 11: Schreibe ein Programm mit einer Rechteckprozedur, und zeichne in diesem Programm ein Bungalow.

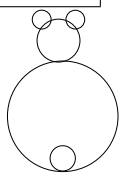
Aufgabe 12: Schreibe eine Prozedur, die ein Zehneck mit einer beliebigen Seitenlänge zeichnet, und rufe diese Prozedur im Hauptprogramm mehrmals auf.



Aufgabe 13: Schreibe eine `procedure vieleck(anzahl,seite: Integer);` die ein Vieleck zeichnet mit der Seitenlänge `seite`. Die Anzahl der Eckpunkte wird im Parameter `anzahl` übergeben. Der Drehwinkel hängt von der Anzahl der Eckpunkte ab. Dazu musst du wissen, wie man in Pascal ganze Zahlen dividiert.

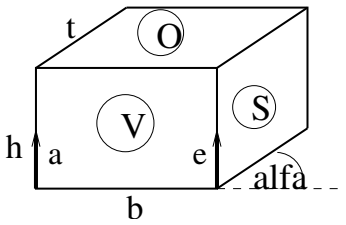
50 DIV 10 ergibt 5, und 50 DIV 12 ergibt 4, der Rest fällt unter den Tisch. Man darf überall, wo man eine Zahl hinschreiben darf, auch einen Rechenausdruck hinschreiben, dessen Ergebnis eine Zahl ist, also z.B.: forwd(12 DIV 4)

Aufgabe 14: Nutze die Prozedur aus der vorigen Aufgabe, um eine Figur aus unterschiedlich großen Kreisen zu zeichnen.



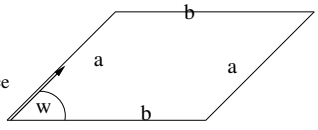
Geschachtelte Prozeduren, Quader

In diesem Abschnitt beruht alles auf Quadern. Das Schrägbild eines Quaders besteht aus drei Parallelogrammen, nämlich V, O und S. Die Form des Quaders ist bestimmt durch die Längen h, b, t (für Höhe, Breite, Tiefe), und den Schrägbildwinkel alfa, der allerdings innerhalb des ganzen Programmes gleich ist. Deshalb solltest du diesen Winkel als in der dritten Zeile des Programms als Konstante festlegen, also nach dem `uses rlturtle;` kommt `const alfa=30;`



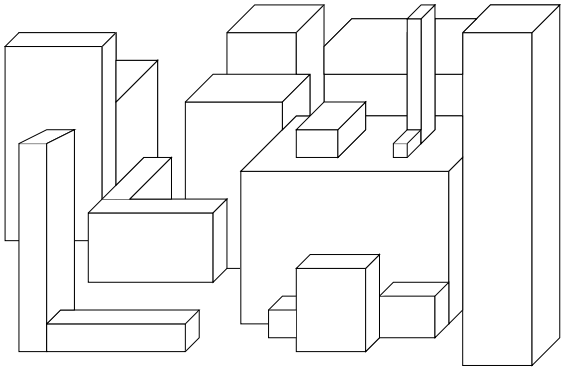
Der Programmkopf der Quaderprozedur lautet also: `procedure quader(h,b,t: Integer);` Die Quaderprozedur ruft nun dreimal eine Parallelogrammprozedur auf: V hat die Maße h und b und den Winkel 90°, O hat die Seiten t und b und den Winkel alfa, und S hat die Maße h und t und den Winkel (90°-alfa).
Man braucht also zuerst eine Parallelogrammprozedur, der man zwei Seiten und einen Winkel angeben kann.

Aufgabe 15: Schreibe diese Parallelogrammprozedur entsprechend der nebenstehenden Skizze, und dann darauf aufbauend die Quaderprozedur.



Die folgenden Aufgaben verwenden die Quaderprozedur. Oft möchte man, dass die Quader sich gegenseitig überdecken. Das gelingt, wenn man in einem BIld die Quader die hinteren Quader zuerst zeichnet und danach die vorderen, und wenn jeder Quader beim Zeichnen seinen Hintergrund löscht.

Dass geschieht, wenn du als erste Zeile der Parallelogrammprozedur schreibst `polyclr;` und als letzte Zeile `polyend;` Wenn polyclear aufgerufen wird, merkt sich die turtle alle Ecken, und bei polyend löscht sie das Innere des durch diese Ecken beschriebenen Vielecks.



Aufgabe 16: Verwende die Quaderprozedur, um eine moderne Stadtszenerie zu zeichnen.

Aufgabe 17: Zur Erinnerung an Schleifen kannst du nun die beiden Quaderreihen zeichnen.