

## Das Arbeiten mit Listen

MuPAD stellt Mengen, Folgen und Listen zum Zusammenfassen von Elementen zur Verfügung. Listen sind vor allem für das Programmieren ein gutes und notwendiges Hilfsmittel, wie später zu sehen sein wird.

Eine Liste ist eine geordnete Folge beliebiger MuPAD-Objekte, die in eckige Klammern eingeschlossen wird.

- **Liste := [a, 5, sin(x)^2 + 4, [a, b, c], hallo, 3/4, 3.9087];** (*[... AltGr + 8; ]... AltGr + 9*)

In einer Liste können beliebige Objekte stehen. Wie [a, b, c] im Beispiel zeigt, kann eine Liste selbst wiederum Listen als Elemente enthalten. Listen können auch leer sein:

- **LeereListe := [ ];** definiert eine leere Liste

Der  $\$$ -Operators zur automatischen Erzeugung langer Folgen ist sehr hilfreich zur Konstruktion langer Listen:

- **Folge := n \$ n = 1..20; Liste1 := [Folge];**

Der erste Befehl definiert eine Folge natürlicher Zahlen, die dann zur Konstruktion der Liste herangezogen wird. Dies geht natürlich auch direkt, wie die folgenden drei Beispiele zeigen:

- **Liste1 := [n \$ n = 1..20]; Liste2 := [2^n \$ n = 1..20]; Liste3 := [x^i \$ i = 1..20];**

Die Anzahl der Elemente einer Liste kann mit der Funktion *nops* festgestellt werden, die Elemente können mit der Funktion *op* ausgelesen werden.

- **nops(Liste); nops(Liste1);** Anzahl der Elemente der ersten Liste: 7; liefert 20
- **op(Liste, 5); op(Liste1, 10);** liefert 5. Element der Liste *Liste*: hallo; liefert 10
- **op(Liste, 2..4); op(Liste);** 2., 3. und 4. Element der Liste *Liste*; alle Elemente der Liste *Liste*

Eine alternative Möglichkeit, einzelne Listenelemente zu erhalten, liefert der *Index*-Operator, der schneller auf die Listenelemente zugreift, als es die *op*-Funktion vermag. (Rechenzeit bei Verwendung von Schleifen!)

- **Liste[1]; Liste[6];** erstes Element der Liste *Liste* bzw. sechstes Element der Liste *Liste1*.

Man kann den Wert eines Elements einer Liste verändern:

- **Liste[1] := neu; Liste;**

Das erste Element der Liste *Liste* war vorher mit *a* belegt. Nun wurde diesem Element der Wert *neu* zugewiesen.

Das Entfernen eines Elements aus einer Liste erfolgt durch *unassign*:

- **unassign(Liste[1]); Liste; nops(Liste);** entfernt das erste Element  $\Rightarrow$  *Liste* enthält nur 6 Elemente

Mit der Funktion *append* können Elemente an eine Liste angehängt werden:

- **append(Liste, neu1);** hängt das Element *neu1* am Ende der Liste an, verändert sie nicht dauerhaft.
- **Liste; nops(Liste);** Die Liste enthält immer noch die alten 6 Elemente.

Man kann aber die durch *append* neu entstehende Liste einer anderen Liste oder derselben Liste zuweisen:

- **Liste3 := append(Liste, neu1);** neue Liste *Liste3* mit dem angehängten Element ist entstanden
- **Liste; nops(Liste); Liste3; nops(Liste3);**
- **Liste := append(Liste, neu1);** Die Liste *Liste* wurde nun dauerhaft verändert.
- **Liste; nops(Liste);** Die Liste *Liste* besteht nun wieder aus 7 Elementen.

Es ist möglich, gleichzeitig mehrere Elemente anzuhängen:

- **Liste := append(Liste, neu2, neu3, neu4); Liste; nops(Liste);**

Mit Hilfe des *Punkt*-Operators können zwei Listen zusammengefügt werden. (Reihenfolge der Listen beachten!)

- **Liste1.Liste3; Liste3.Liste1;**

Mit *sort* werden Listen sortiert. Numerische Werte werden ihrer Größe nach, Zeichenketten lexikographisch angeordnet:

- **L1 := [-1.23, 4, 3, 2, 1/2]; sort(L1);** sortiert die neu definierte Liste *L1*
- **L2 := ["A", "c", "abc", "a1", "aa", "C", "Abc"]; sort(L2);** sortiert die neu definierte Liste *L2*
- **L3 := [A, c, abc, a1, aa, C, Abc]; sort(L3);** sortiert die neu definierte Liste *L3*

Man beachte, dass die lexikographische Anordnung nur bei Benutzung von mit " erzeugten Zeichenketten verwendet wird (z.B. *L2*). Bei Namen von Bezeichnern wird u.a. die Länge der Namen berücksichtigt (siehe *L3*).

Die Funktion *select* dient dazu, Listenelemente mit bestimmten Eigenschaften aus einer Liste herauszufiltern. Die Funktion *select* liefert eine Liste mit allen Elementen, die die angegebene Eigenschaft erfüllen:

- **L4 := [n \$ n=1..20]; select(L4, isprime);** liefert alle Primzahlen der Liste *L4*

### Aufgaben:

1. Erstellen Sie eine Liste L1, welche die ersten 100 Quadratzahlen enthält. Nutzen Sie den  $\$$ -Operator.
2. Erstellen Sie eine Liste L2, die 20 Kubikzahlen enthält, angefangen bei der 11. Kubikzahl.
3. Hängen Sie an die Liste L2 die Zahl 3333 an (dauerhaft verändern!).
4. Fügen Sie die beiden Listen L1 und L2 zur Liste L3 zusammen und sortieren sie die Liste L3.
5. Überschreiben Sie das dritte Element mit 10 und entfernen Sie das 13. Element der Liste L3.

Lassen Sie sich jeweils die Ergebnisse (Liste und Anzahl der Elemente) anzeigen.

Speichern Sie unter **Listen.txt** ab.