Übersicht

Lehrinhalt: Programmierung in C

- Überblick über Programmiersprachen, Allgemeines
- C-Basisdatentypen, Zahlendarstellung, Variablen, Konstanten
- Operatoren und Ausdrücke
- Anweisungen
- Kontrollstrukturen
- Funktionen
- Zeiger und Felder
- **Zeichenketten (Strings)**
- Benutzerdefinierte Datentypen
- Dynamischer Speicher
- Dateiarbeit
- Funktionspointer, Rekursion
- Preprozessor

Zeichenketten (Strings)

- Programme verarbeiten neben numerischen Daten oft auch Daten in Textform.

- Texte sind Zeichenketten, die auch als Strings bezeichnet werden

- In C gibt es keinen eigenständigen Datentyp für Zeichenketten.

- Zeichenketten werden in C als Felder einzelner Zeichen (char) gehandhabt. Jedes Element als Zeichen ist im ASCII-Code kodiert.

Zeichenketten (Strings)

Eine Zeichenkette muss in C als Feld deklariert werden.
Beispiel: `char text[10];  // 10 Zeichen, davon 9 benutzbar`

Es ist dann möglich, einzelne Zeichen auf die Feldelemente zuzuweisen.

```
text[0] = 'H';
text[1] = 'T';
text[2] = 'W';
text[3] = '\0';
```

Es ist auch möglich, einzelne Zeichen auszuwählen und zu verarbeiten.

```
if (text[0]=='H' && text[1]=='T' && text[2]=='W')
        printf("HTW im Text gefunden.\n");
```

Das erscheint ziemlich umständlich.

Zeichenketten (Strings)

Anstelle umständlicher Zugriffe auf einzelne Zeichen werden

- Zeichenkettenkonstante benutzt, um Initialisierungen vorzunehmen, zum Beispiel:

```
char text[10] = "HTW";
```

- Funktionen aus der Standardbibliothek benutzt, zum Anhängen, zum Kopieren, zum Vergleich, zur Eingabe und zur Ausgabe.
  Beispiel: `strcat(text, " DD"); //ergibt "HTW DD" auf text`

- Funktionsargumente als Zeichenkettenkonstante angegeben.
  Beispiel:

```
if ( strcmp(text, "HTW DD" )==0 )
  printf("An der %s muss man fleissig sein!\n", text);
```

Zeichenketten: Endekennzeichen

Vereinbarung:
Das Ende der Zeichenkette wird mit den Zeichen ‘\0‘ gekennzeichnet.
Die Länge der Zeichenkette muss deshalb im Vergleich zu numerischen Feldern nicht extra verwaltet werden.

Damit verbunden:

- Bei Zeichenkettenkonstanten erzeugt der Übersetzer immer automatisch ein Nullzeichen ‘\0‘ am Ende.
- Die Standardbibliotheksfunktionen setzen die Endekodierung mit ‘\0‘ voraus. Sie erzeugen auch Zeichenketten als Ergebnis, die mit ‘\0‘ abgeschlossen sind (s.g. nullterminiert).
- Es müssen ausreichend Elemente in der Zeichenkette bei der Deklaration als Feld angegeben werden. Es ist auch Platz für das abschließende ‘\0‘-Zeichen vorzusehen.

Zeichenketten: Zuweisung

Zuweisung von Zeichenketten:

Anstelle des Zuweisungsoperators = ist die Funktion strcpy() zu nutzen.

```
strcpy(nachricht, "Auf Wiedersehen!");

// nachricht = "Auf Wiedersehen!"; funktioniert nicht
```

Zum Vergleich: Das Kopieren von Feldern gemäß `feldB = feldA;` ist in C auch nicht sinnvoll, da feldA und feldB als Zeiger verstanden werden. Man würde damit dem Zeiger feldB überschreiben.

```
int feldA[3], feldB[3] = { 11, 22, 33 };
int i, n = 3;

// feldA = feldB;       // funktioniert nicht,
                        // Compiler meldet Fehler
for (i = 0; i < n; i = i + 1)
  feldA[i] = feldB[i]; // das funktioniert
```

Zeichenkettenkonstante

Zeichenkettenkonstante werden in Doppel-Apostroph eingeschlossen:

```
"Das ist eine nette Zeichenkette!"
```

Nutzung für eine Initialisierung bei Deklaration:

```
char zk[40] = "Das ist eine nette Zeichenkette!";
// ergibt: zk[40]={'D', 'a', 's', ' ', 'i', 's', 't', …,
                   'k', 'e', 't', 't', 'e', '!', '\0'}
```

Eine solche Zuweisung ist nur in Zusammenhang mit der Deklaration möglich! Sonst muss die Zuweisung über die Funktion *strcpy*() erfolgen.

Zeichenketten: Ausgabe

Ausgabe von Zeichenketten

- **über die printf()-Funktion**

```
printf("Ausgabe: %s", zk);
```

%s (string) ist das Format für Zeichenkette.
Die übergebene Zeichenkette (hier zk) muss durch '\0' abgeschlossen sein!

- **per puts()-Funktion:**

```
char ausgabe[50]="Start in 10 Sekunden.";
int rc;

rc= puts(ausgabe);

if (rc==EOF)
    printf("Fehler bei der Ausgabe\n");
```

Zeichenketten: Eingabe

Eingabe von Zeichenketten

Kann über die scanf()-Funktion erfolgen:

```
scanf("%10s", zk);
```

- s (string) ist das Format-Umwandlungszeichen für eine Zeichenkette;
- 10 ist die maximale Länge der zu übernehmenden Zeichen
- Eingabe wird durch ENTER abgeschlossen!
  Achtung: Ein eingegebenes Leerzeichen schließt bei %s die zu übernehmenden Zeichen ab!

Eingabe aller Zeichen (inkl. Leerzeichen):

```
scanf("%[^\n]", zk); // liest bis zum Zeilenumbruch ein
```

Zeichenketten: Eingabe

Eingabe von Zeichenketten (Fortsetzung)

Mit der gets()-Funktion (get string):

```
char eingabe[50];
char *s;

s=gets(eingabe);  // Rückgabe des Zeigers eingabe
                  //bei Erfolg

if (s==NULL)
      printf("Fehler bei der Eingabe\n");
```

Noch sicherer funktioniert fgets():

```
#define LEN 50
char eingabe[LEN];
char *s;
s = fgets(eingabe, LEN, stdin); // Liest nur 50 Zeichen
                                // Nullzeichen eingeschl.
```

Standardfunktionen für Zeichenketten

Voraussetzung: `#include <string.h>`

Kopieren / Zuweisen strcpy()

```
strcpy(ziel, quelle);
```

Kopiert die Zeichenkette des char-Feldes quelle auf das char-Feld ziel und erzeugt ein Endezeichen ‘\0‘. Der Rückkehrwert von strcpy ist vom Typ char *, d.h. ein Zeiger auf die Zeichenkette ziel.

Verketten von Zeichenketten strcat()

```
strcat(ziel, quelle);    // ziel=ziel°quelle
```

Kettet die Zeichenkette des char-Feldes quelle an das char-Feld ziel. Der Rückkehrwert von strcat ist vom Typ char *, d.h. ein Zeiger auf die Zeichenkette ziel. Damit lässt sich auch ein Funktionsaufruf wie folgt benutzen:

```
strcpy(text1, strcat("Das ist", " eine Verkettung"));
```

Standardfunktionen für Zeichenketten

Vergleichen von Zeichenketten strcmp()

```
int ergebnis = strcmp(text1, text2);
```

Vergleicht die Zeichenkette des char-Feldes text1 mit der Zeichenkette des char-Feldes text2 lexikographisch.
Der Rückkehrwert von strcmp ist vom Typ int.

Ergebnis :

- < 0 : text1 kleiner als text2
- ==0 : text1 identisch zu text2
- > 0 : text1 größer als text2

Prinzip:
Verglichen wird anhand des Zahlencodes der Zeichen im ASCII-Code.

Standardfunktionen für Zeichenketten

Länge einer Zeichenkette strlen()

```
int Laenge;

Laenge=strlen(kette);
```

Der Rückkehrwert ist die Länge der Zeichenkette, d.h. die Anzahl der Zeichen ohne das Abschlusszeichen ‘\0‘.

In der Standardbibliothek string.h gibt es noch zahlreiche weitere Funktionen zur Handhabung von Zeichenketten.

Standardfunktionen für Zeichenketten

Länge einer Zeichenkette strlen()

```
int Laenge;

Laenge=strlen(kette);
```

Der Rückkehrwert ist die Länge der Zeichenkette, d.h. die Anzahl der Zeichen ohne das Abschlusszeichen ‘\0‘.

In der Standardbibliothek string.h gibt es noch zahlreiche weitere Funktionen zur Handhabung von Zeichenketten.

Standardfunktionen für Zeichenketten

Weitere nützliche Funktion: `sscanf()`

sscanf() liest formatiert von einer Zeichenkette (anstatt von der Konsoleneingabe) ein und konvertiert die gelesenen Werte in die Zielvariablen.

Beispiel: Einlesen einer Zeichenkette, Prüfen und Übernahme auf Variable für Stunde und Minute

```
char zeile[LEN];    // LEN per #define auf 10 gesetzt
unsigned int i, n;
int h, m;

fgets(zeile, LEN, stdin);  // Einlesen

// Prüfen und Fehlermeldung,  siehe vollst. C-Programm

// Übernahme
n= sscanf(zeile, "%d:%d", &h, &m);
```