Die printf()-Funktion

Nachdem die printf()-Funktion zur Ausgabe auf den Bildschirm schon mehrfach verwendet wurde, soll diese Funktion nun ausführlicher beschrieben werden.

Das allgemeine Format dieser Funktion ist

```
int printf(char* format, argument1, argument2, ...);
```

Die Typangabe `int` besagt, dass die Funktion einen Wert liefert (Rückgabewert). Der Rückgabewert ist die Anzahl der ausgegebenen Zeichen.

In den Klammern werden der Funktion ein oder mehrere Parameter übergeben (Übergabeparameter). Der erste Paramater ist immer der so genannte Formatstring[1]. Der Formatstring ist eine Zeichenkette, welche in Anführungszeichen ("...") stehen muss. Im einfachsten Fall ist das der Text, welcher ab der aktuellen Cursorposition am Bildschirm ausgegeben wird.

Beispiel:

```
printf("Hallo!");
```

Außer den normalen Zeichen kann der Formatstring Formatangaben und/oder Escape-Sequenzen enthalten.

Formatangaben sind erforderlich, wenn weitere Paramater übergeben werden, welche nach dem Formatstring getrennt durch Kommata angegeben werden. Formatangaben werden mit einem %-Zeichen eingeleitet. Beispielsweise führt die Formatangabe `"...%i..."` dazu, dass eine ganze Zahl ausgegeben wird. Das Argument wird an der Stelle ausgegeben, wo im Formatstring das %-Zeichen erscheint.

Beispiel:

```
printf("Die Zahl %i ist eine Primzahl!",13);
```

Ausgabe:

```
Die Zahl 13 ist eine Primzahl!
```

Es können innerhalb des Formatstrings mehrere Formatangaben erscheinen. Die Ausgabe der einzelnen Argumente erfolgt in der Reihenfolge der Formatangaben. Das heißt: Die erste Formatangabe steht als Platzhalter für argument1, die zweite Formatangabe als Platzhalter für argument2, usw... .

Ist das Argument ein mathematischer Ausdruck, wird das rechnerische Ergebnis ausgegeben.

[1] Genauer: Ein Zeiger auf eine Zeichenkette

Beispiel:

```
printf("%i mal %i ist %i.",3,4,3*4);
```

Ausgabe:

```
3 mal 4 ist 12.
```

Als Argumente sind folgende Typen möglich:

- Ganze positive oder negative Zahlen
- Fließkommazahlen vom Typ float
- Fließkommazahlen vom Typ double
- Einzelne Zeichen (im ASCII-Code)
- Texte (Strings)

Die folgende Tabelle zeigt die möglichen Formatangaben auf[1]:

Formatangabe	*Typ des Arguments*
%i oder %d	positive oder negative ganze Zahl
%f oder %lf	Fließkommazahl vom Typ float oder double
%e oder %E	Fließkommazahl, exponentiale Darstellung
%u	vorzeichenlose Ganzzahl
%p	Adressen (Pointer)
%c	einzelnes Zeichen im ASCII-Code
%s	Text (String)
%x oder %X	ganze Zahl, hexadezimale Darstellung
%%	%-Zeichen ausgeben

[1] Die Tabelle erhebt keinen Anspruch auf Vollständigkeit

Das folgende Programm deklariert verschiedene Objekte und demonstriert die Ausgaben bei verschiedenen Formatangaben. Das Ende der Formatstrings beinhaltet die Escape-Sequenz `"...\n"` um einen Zeilenvorschub durchzuführen (siehe unten!).

```
#include<stdio.h>
#include<conio.h>
int main()
{
	int z1=44;
	int z2=-360;
	float z3=1.4142f;
	double z4=3141.25;
	char c=76;
	char text[5]={65,66,67,68,0};
	printf("z1: %i, %d, %u, %x, %X, Laenge in Byte: %i\n",z1,z1,z1,z1,z1,sizeof(z1));
	printf("z2: %i, %d, %u, %x, %X, Laenge in Byte: %i\n",z2,z2,z2,z2,z2,sizeof(z2));
	printf("z3: %f, %lf, %e, %E, Laenge in Byte: %i\n",z3,z3,z3,z3,sizeof(z3));
	printf("z4: %f, %lf, %e, %E, Laenge in Byte: %i\n",z4,z4,z4,z4,sizeof(z4));
	printf(" c: %c, %x, %X, Laenge in Byte: %i\n",c,c,c,sizeof(c));
	printf("Text: %s\n",text);
	_getch();
}
```

Ausgabe:

```
z1: 44, 44, 44, 2c, 2C, Laenge in Byte: 4
z2: -360, -360, 4294966936, fffffe98, FFFFFE98, Laenge in Byte: 4
z3: 1.414200, 1.414200, 1.414200e+000, 1.414200E+000, Laenge in Byte: 4
z4: 3141.250000, 3141.250000, 3.141250e+003, 3.141250E+003, Laenge in Byte: 8
 c: L, 4c, 4C, Laenge in Byte: 1
Text: ABCD
```

Mit zusätzlichen Formatanweisungen lässt sich die Länge einer Ausgabe verändern. Das erfolgt im wesentlichen dadurch, dass nach dem %-Zeichen die gewünschte Länge angegeben wird. Beispielsweise wird mit der Formatangabe `"...%5i..."` eine ganze Zahl rechtsbündig mit 5 Stellen ausgegeben. Mit der Formatangabe `"...%05i..."` werden führende Stellen mit 0 aufgefüllt.

Beispiel:

```
#include<stdio.h>
#include<conio.h>
int main()
{
	int z1=44;
	printf("z1: %5i\n",z1);
	printf("z1: %05i\n",z1);
	_getch();
}
```

Ausgabe:

```
z1:    44
z1: 00044
```

Bei der Längenangabe von Fließkommazahlen muss die Gesamtlänge der Zahl und die Anzahl der Nachkommastellen angegeben werden. Die Gesamtlänge ergibt sich aus der Anzahl der Vorkammastellen, dem Dezimalpunkt und der Anzahl der Nachkommastellen. Beispielsweise die Formatangabe "...%8.3f..." bewirkt die Ausgabe einer Zahl mit 4(!) Vorkomma- und drei Nachkommastellen. Weitere Nachkommastellen werden gerundet. Mit einem zusätzlichen '-' Zeichen unmittelbar nach dem %-Zeichen lassen sich Zahlen und Strings auch linksbündig ausgeben.

Beispiel für Zahlenausgabe:

```
#include<stdio.h>
#include<conio.h>
int main()
{
     double z1=3.141592;
     printf("z1: %8.3f\n",z1);
     printf("z1: %-8.3f\n",z1);
     printf("z1: %08.3f\n",z1);
     _getch();
}
```

Ausgabe:

```
z1:    3.142
z1: 3.142
z1: 0003.142
```

Beispiel für Textausgabe:

```
#include<stdio.h>
#include<conio.h>
int main()
{
     printf("%10s\n","Hallo");  //rechtsbündige Ausgabe
     printf("%-10s\n","Hallo"); //linksbündige Ausgabe
     _getch();
}
```

Ausgabe:

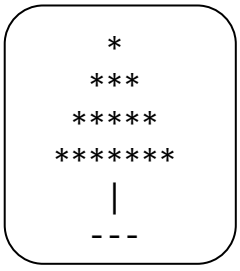
```
     Hallo
Hallo
```

Neben den darstellbaren Zeichen können im Formatstring Steuerzeichen eingefügt werden (sogenannte Escape-Sequenzen). Escape-Sequenzen beginnen mit einem '\'-Zeichen (Backslash). Beispielsweise bewirkt die Formatanweisung `"...\n..."`, dass der Cursor an den Anfang der nächsten Zeile gesetzt wird. Weitere Escape-Sequenzen sind in der folgenden Tabelle dargestellt[1].

Formatangabe	*Bedeutung*
\n	setzt den Cursor an den Anfang der nächsten Zeile
\t	setzt den Cursor auf die nächste horizontale Tabulatorposition
\r	setzt den Cursor an den Anfang der Zeile
\b	setzt den Cursor um eine Position zurück
\a	gibt einen Alarmton aus (alert)
\\	das Zeichen \ wird ausgegeben
\"	das Zeichen " wird ausgegeben

Übung:

Auf dem Bildschirm soll folgendes Muster ausgegeben werden:



Entwickeln Sie bitte ein entsprechendes Programm! Das Programm soll so gestaltet werden, dass in den Stringkonstanten keine Leerzeichen (Space) verwendet werden.

[1] Die Tabelle erhebt keinen Anspruch auf Vollständigkeit. Außerdem ist das Verhalten u.U. systemabhängig.

Die scanf_s()-Funktion

Die scanf_s()-Funktion wird u.a. für die Eingabe von Zahlenwerten über die Tastatur verwendet.

Das allgemeine Format dieser Funktion ist

```
int scanf_s(char* format, argument,...);
```

Ähnlich wie die printf()-Funktion ist der erste Übergabeparameter ein Formatstring, welcher den Typ des einzugebenden Wertes beschreibt. Einige Möglichkeiten beschreibt die folgende Tabelle:

Formatangabe	*Typ des Arguments*
%i	ganze Zahl
%f	Fließkommazahl vom Typ float
%lf	Fließkommazahl vom Typ double

Der weitere Übergabeparameter ist die (Speicherplatz-)Adresse des einzugebenden Objektes.
Die Adresse eines Objektes erhält man, indem dem Bezeicher ein &-Zeichen vorangestellt wird (Referenzierungsoperator).

Im folgenden Beispiel werden drei Zahlen vom Typ int, float und double eingegeben und danach angezeigt. Eine Eingabe schließt mit der Return-Taste ab.

```
#include<stdio.h>
#include<conio.h>
int main()
{
      int z1=0;
      float z2=0;
      double z3=0;
      printf("Zahl1: ");
      scanf_s("%i",&z1);
      printf("Zahl2: ");
      scanf_s("%f",&z2);
      printf("Zahl3: ");
      scanf_s("%lf",&z3);
      printf("1.Zahl: %i\n",z1);
      printf("1.Zahl: %f\n",z2);
      printf("1.Zahl: %lf\n",z3);
      _getch();
      return(0);
}
```

Mögliche Bildschirmausgabe:

```
Zahl1: 234
Zahl2: 3.1415
Zahl3: 2.3
1.Zahl: 234
1.Zahl: 3.141500
1.Zahl: 2.300000
```

Hinweis:

Das Programm fängt keine Fehler auf, welche durch falsche Eingaben des Benutzers entstehen! Beispielsweise kann nicht verhindert werden, dass der Benutzer fälschlicherweise Kommata oder Buchstaben eingibt. Um sichere Eingaben zu realisieren sind benutzerdefinierte Funktionen zu entwickeln, welche in späteren Kapiteln beschrieben werden.

Die _getch()-Funktion

Die _getch()-Funktion wird verwendet um ein einzelnes Zeichen über die Tastatur einzugeben.

Das allgemeine Format dieser Funktion ist

```
int _getch();
```

Der Rückgabewert der Funktion ist der ASCII-Code der gedrückten Taste. Der Wert kann einem Objekt vom Typ `char` oder `int` zugewiesen werden.

Beispiel:

```
...
int c;
c=_getch();
...
```

Das folgende Programmbeispiel liest fortlaufend ein Zeichen von der Tastatur ein und gibt es auf dem Bildschirm aus bis die ESC-Taste (ASCII_Code 27) betätigt wird.

```
#include<stdio.h>
#include<conio.h>
int main()
{
      int z=0x00;
      do
      {
      z=_getch();
      if(z==27) break;      //Schleife verlassen, wenn ESC betätigt
      printf("%c",z);
      }
      while(1);             //wiederhole immer
      return(0);
}
```