Merkblatt Zeiger(Pointer)

Ein Zeiger ist eine Größe, deren Inhalt die Adresse
- -eines Objekts eines bestimmten Typs ist (typisierte Zeiger).
- -eines Objekts eines beliebigen Typs ist (void Zeiger).

Beispiel:
```
int *ip; /* Zeiger auf int-Objekte */
void *vp; /* Zeiger auf beliebige Objekte */
```

Spezielle Konstante:

Der Nullzeigerwert `(void *)0` oder ***NULL*** zeigt nirgends hin.
Er ist die einzige konstante Adresse, die systemunabhängig
- -einem Zeiger zugewiesen werden kann.
- -mit dem Inhalt eines Zeigers auf (Un-)Gleichheit getestet werden kann.

Operationen/Funktionen im Zusammenhang mit Zeigern:

`*`	Dereferentierung
`&`	Adressbildung
`+`	Addition eines int Wertes, falls der Zeiger auf ein Array zeigt
`-`	Subtraktion eines int Wertes, falls der Zeiger auf ein Array zeigt
	Subtraktion zweier Zeiger, falls beide auf das gleiche Array zeigen
`!= ==`	Test auf (Un-)Gleichheit zweier Zeiger
`< <= >= >`	Vergleichstest zweier Zeiger, die auf das gleiche Array zeigen
`malloc`	Bereitstellen von zusammenhängendem Speicher zur Laufzeit
`realloc`	Vergrößern/Verkleinern von Speicher zur Laufzeit
`free`	Freigabe von Speicher zur Laufzeit

Zusammenhang zwischen Zeigern und Datenstrukturen:

`T a[D];` **`a[d] ≡ *(a+d)`**

oder allgemein

`T a[D1][D2]..[Dn];` **`a[d1][d2]..[dn] ≡ *(*(..*(a+d1)+d2)+..dn)`**

Dabei ist a die Adresse der ersten Komponente `a[0][0]..[0]`.
`a[d1]` ist die Adresse der d1. Komponente eines Arrays mit [D2]..[Dn] Komponenten und n-1 Dimensionen.
Die Adresse von `a[d1]` entspricht `a+d1*D2*D3*..Dn*sizeof(T).`

Beispiele:
`int a[5][4]; /* Annahme: sizeof(int) sei 4 und &a sei 1000 */`

Ausdruck	Wert	Typ
a	1000	int[5][4] ≡ int **
a[i]≡*(a+i*4*sizeof(int))	1000+i*16	int[4] ≡ *int* *
a[i][j]≡*(*(a+i*4*sizeof(int))+j*sizeof(int))	Wert von a[i][j]	int

```
struct {... T m; ...} s,*sp;
sp = &s;
```

```
s.m   ≡ (&s)->m

sp->m ≡ (*sp).m
```