5.1.F Einführung und Klassen/Neue Möglichkeiten für structs – Ergänzungen und Bilder

5.1.F.1 Klasse mit Methode in C

Auch in C kann man eine Methode innerhalb einer Klasse anlegen, und zwar als Funktionszeiger.

```
#include <stdio.h>
#include <string.h>
struct persontyp
{
       char name[30];
       char vorname[30];
       long gebdat;
       void (*print)(struct persontyp p);
};
void printperson(struct persontyp p)
{
    printf("%s %s, geb %d\n", p.vorname, p.name, p.gebdat);
}
int main(void)
{
    struct persontyp chef={"Meier","Heinz",19501212,printperson};
    chef.print(chef);
    return 0;
}
```

Das Beispiel erlaubt sogar Polymorphie, denn man kann diesen Funktionszeiger `print` bei verschiedenen Objekten auf verschiedene Funktionen nach Art von `printperson` zeigen lassen, so dass unterschiedliche Personenarten (Kunden, Lieferanten) unterschiedliche Anzeigen bekommen.

Unschön bleibt in C die doppelte Nennung des Objekts `chef` im Methodenaufruf. Durch Makros kann man sie wieder reduzieren:

```
#include <stdio.h>
#include <string.h>
#define PRINT(x) x.print(x)
struct persontyp
{
       char name[30];
       char vorname[30];
       long gebdat;
       void (*print)(struct persontyp p);
};
void printperson(struct persontyp p)
{
    printf("%s %s, geb %d\n", p.vorname, p.name, p.gebdat);
}
int main(void)
{
    struct persontyp chef={"Meier","Heinz",19501212,printperson};
    PRINT(chef);
    return 0;
}
```

Das Ergebnis sieht dann aber wieder typisch nach C aus, was allerdings nur ein kosmetisches Problem ist.