Es gibt grundsätzlich zwei Arten von Anweisungsblöcken, die gerne "Funktionen" genannt werden.

Zum einen wären da die *Prozeduren*.

Eine *Prozedur* führt eine Aufgabe aus, hat aber keinen Rückgabewert.

Ein Beispiel für eine Prozedur wäre soetwas:

void SchreibeZahl(int Zahl);

Diese Prozedur würde die Zahl auf dem Bildschirm ausgeben, die ihr als Parameter übergeben wurde.

Eine *Funktion* verfügt hingegen über einen Rückgabewert und sollte nach Möglichkeit NUR diesen Rückgabewert übergeben.

Tut eine Funktion noch andere Dinge (Beispielsweise die Bearbeitung einer Variable oder eine Ausgabe auf dem Bildschirm), so ist das unsauberer Code, der in größeren Projekten schnell für Verwirrung sorgen kann.

Grundsätzlich sollten Funktionen und Prozeduren klar abgegrenzte Funktionalität haben und über keine derartigen Nebenwirkungen verfügen.

Ein Beispiel für eine saubere Funktion hatten wir bereits im Unterricht:

int max(int Nummer1, int Nummer2);

Diese Funktion gibt einfach als Rückgabewert aus, welche der beiden Zahlen höher ist.

Wie ein Rückgabewert funktioniert:

int Zahl;

Zahl = max(5,7);

printf("%d", Zahl);

printf("%d", max(Zahl, 10));

Dies wäre gültiger Code. Da der Rückgabewert von max() ein Integer ist, kann man diese Funktion behandeln, als wäre sie eine stinknormale Ganzzahl.

Allerdings ist diese Trennung zwischen Funktion und Prozedur meist nicht so sauber wie hier dargestellt.

Beispiel:

int printf (const char \*format, ...);

int scanf ( const char \* format, ... );

Scanf und printf geben beide Rückgabewerte aus, obwohl sie bereits andere Dinge tun.

Das heißt, dass folgender Code gültig ist:

printf("%d", printf("Hello World!"));

Ausgabe:

Hello World!12

Der Rückgabewert von printf ist die Menge an Zeichen, die ausgegeben wurden.

Bei scanf wird ausgegeben, wieviele gültige Zuweisungen stattgefunden haben.

Will man also testen, ob jemand wirklich eine Zahl und keinen Buchstaben eingegeben hat, kann man das so tun:

int Zahl;

printf("Geben sie eine Zahl ein!");

if(scanf("%d", Zahl)) {

printf("Glueckwunsch, sie haben eine Zahl eingegeben!");

}

else {

printf("Dies war definitiv keine Zahl.");

}

Der Grund dafür ist, dass die if-Bedingung Boolsche Werte auswertet. In C bedeutet das, dass eine 0 falsch ist und alles andere wahr. Gibt scanf also eine 0 zurück, weil keine gültigen Zuweisungen stattgefunden haben da man keine Zahl eingegeben hat, so ist die if-Bedingung falsch und springt zum Else.

Leider funktioniert die Zuweisung auch, wenn man eine Fließkommazahl eingibt, obwohl der Datentyp eigentlich Integer ist.